Contracting preference relations for database applications[☆]

Denis Mindolin, Jan Chomicki

Department of Computer Science and Engineering
201 Bell Hall, University at Buffalo
Buffalo, NY 14260-2000, USA

Abstract

The binary relation framework has been shown to be applicable to many real-life preference handling scenarios. Here we study preference contraction: the problem of discarding selected preferences. We argue that the property of minimality and the preservation of strict partial orders are crucial for contractions. Contractions can be further constrained by specifying which preferences should be protected. We consider two classes of preference relations: finite and finitely representable. We present algorithms for computing minimal and preference-protecting minimal contractions for finite as well as finitely representable preference relations. We study relationships between preference change in the binary relation framework and belief change in the belief revision theory. We also introduce some preference query optimization techniques which can be used in the presence of contraction. We evaluate the proposed algorithms experimentally and present the results.

Key words: preference contraction, preference change, preference query

1. Introduction

A large number of preference handling frameworks have been developed [Fis70, BBD+04, HGY06]. In this paper, we work with the *binary relation* preference framework [Cho03, Kie02]. Preferences are represented as binary relations over tuples. They are required to be *strict partial orders (SPO)*: transitive and irreflexive binary relations. The SPO properties are known to capture the rationality of preferences [Fis70]. This framework can deal with finite as well as infinite preference relations, the latter represented using finite *preference formulas*.

Working with preferences in any framework, it is naive to expect that they never change. Preferences can change over time: if one likes something now, it does not mean one will still like it in the future. Preference change is an active topic of current research [Cho07a, Fre04]. It was argued [Doy04] that along with the discovery of sources of preference change and

[☆]Research partially supported by NSF grant IIS-0307434. This paper is an extended version of [MC08]

Email addresses: mindolin@cse.buffalo.edu (Denis Mindolin), chomicki@cse.buffalo.edu (Jan Chomicki)

elicitation of the change itself, it is important to preserve the correctness of preference model in the presence of change. In the binary relation framework, a natural correctness criterion is the preservation of SPO properties of preference relations.

An operation of preference change – preference revision – has been proposed in [Cho07a]. We note that when a preference relation is changed using a revision operator, new preferences are "semantically combined" with the original preference relation. However, combining new preferences with the existing ones is not the only way people change their preferences in real life. Another very common operation of preference change is "semantic subtraction" from a set of preferences another set of preferences one used to hold, if the reasons for holding the *contracted* preferences are no longer valid. That is, we are given an initial preference relation $\succ$ and a subset CON of $\succ$ (called here a *base contractor*) which should not hold. We want to change $\succ$ in such a way that CON does not hold in it. This is exactly opposite to the way the preference revision operators change preference relations. Hence, such a change cannot be captured by the existing preference revision operators.

In addition to the fact that discarding preferences is common, there is another practical reason why preference contraction is important. In many database applications, preference relations are used to compute sets of the best (i.e. the most preferred) objects according to user's preferences. Such objects may be cars, books, cameras etc. The operator which is used in the binary relation framework to compute such sets is called *winnow* [Cho03] (or *BMO* in [Kie02]). The winnow operator is denoted as $w_{\succ}(r)$, where r is the original set of objects, and $\succ$ is a preference relation. If the preference relation $\succ$ is large (i.e. the user has many preferences), the result of $w_{\succ}(r)$ may be too narrow. One way to widen the result is by discarding some preferences in $\succ$. Those may be the preferences which do not hold any more or are not longer important.

In this paper, we address the problem of contraction of preference relations. We consider it for finitely representable infinite preference relations (Example 1) and finite preference relations (Example 2).

Example 1. *Assume that Mary wants to buy a car. She prefers newer cars, and given two cars made in the same year, a cheaper one is preferred.*

$$o \succ o' \equiv o.year > o'.year \vee o.year = o'.year \wedge o.price < o'.price$$

where $>, <$ denote the standard orderings of rational numbers, the attribute year defines the year when cars are made, and the attribute price – their price. The information about all cars which are in stock now is shown in the table below:

id	*make*	*year*	*price*
t_1	*VW*	*2007*	*15000*
t_2	*VW*	*2007*	*20000*
t_3	*Kia*	*2006*	*15000*
t_4	*Kia*	*2007*	*12000*

Then the set of the most preferred cars according to $\succ$ is $S_1 = \{t_4\}$. Assume that having observed the set S_1, Mary understands that it is too narrow. She decides that the car t_1 is

not really worse than t_4*. She generalizes that by stating that the cars made in* 2007 *which cost* 12000 *are not better than the cars made in* 2007 *costing* 15000*. So* t_4 *is not preferred to* t_1 *any more, and thus the set of the best cars according to the new preference relation should be* $S_3 = \{t_1, t_4\}$.

The problem which we face here is how to represent the change to the preference relation $\succ$*. Namely, we want to find a preference relation obtained from* $\succ$*, in which certain preferences do not hold. A naive solution is to represent the new preference as* $\succ_1 \equiv\ (\succ - CON)$*, where* $CON(o, o') \equiv o.year = o'.year = 2007 \wedge o.price = 12000 \wedge o'.price = 15000$*, i.e.,* CON *is the preference we want to discard. So*

$$o \succ_1 o' \equiv (o.year > o'.year \vee o.year = o'.year \wedge o.price < o'.price) \wedge \\ \neg(o.year = o'.year = 2007 \wedge o.price = 12000 \wedge o'.price = 15000).$$

However, $\succ_1$ *is not transitive since if we take* $t_5 = (VW, 2007, 12000)$*,* $t_6 = (VW, 2007,$ 14000*), and* $t_7 = (VW, 2007, 15000)$*, then* $t_5 \succ_1 t_6$ *and* $t_6 \succ_1 t_7$ *but* $t_5 \not\succ_1 t_7$*. Hence, this change does not preserve SPO. To make the changed preference relation transitive, some other preferences have to be discarded in addition to* CON*. At the same time, discarding too many preferences is not a good solution since they may be important. Therefore, we need to discard a minimal part of* $\succ_1$ *which contains* CON *and preserves SPO in the modified preference relation. An SPO preference relation which is minimally different from* $\succ_1$ *and does not contain* CON *is shown below:*

$$o \succ_2 o' \equiv (o.y > o'.y \vee o.y = o'.y \wedge o.p < o'.p) \wedge \\ \neg(o.y = o'.y = 2007 \wedge o.p = 12000 \wedge o'.p > 12000 \wedge o'.p \leq 15000)$$

The set of the best cars according to $\succ_2$ *is* $S_2' = \{t_1, t_4\}$*. As we can see, the relation* $\succ_2$ *is different from the naive solution* $\succ_1$ *in the sense that* $\succ_2$ *implies that a car made in* 2007 *costing* 12000 *is not better than a car made in* 2007 *costing* from 12000 to 15000*. We note that* $\succ_2$ *is not the only relation minimally different from* $\succ_1$ *and not containing* CON*.*

Example 2. *Let Mary have the following preferences over cars. She prefers* VW to Kia*. Given two VW, her preference over color is* red is better than green, which is better than blue*. Given two Kias, her preference over color is* green is better red, which is better than blue*. In this example, we use the* ceteris paribus *semantics [BBD+04]: the preference statements above are used to compare only the tuples different in a single attribute. The SPO preference relation* $\succ_1$ *representing Mary's preferences is shown in Figure 1(a). An edge from a tuple to another one denotes the preference of the first tuple to the second one.*

Assume that after some time, Mary decides to change her preferences: a red VW is not better than a red Kia*, and* a green VW is not better than a blue Kia*. That means that the set of preferences we need to drop from the current preference relations is* $CON = \{($ (VW, red) *,* (Kia, red) $); ($ $(VW, green)$*,* $(Kia, blue)$ $)\}$*. The corresponding edges are dashed in Figure 1(a). Clearly, if these edges are removed from the graph, it will not be transitive any more. Hence, additional edges need to be removed to preserve the transitivity of the preference relation. One minimal set of edges whose removal along with* CON *results in a transitive preference relation is shown in Figure 1(b) as the dashed edges.*

(a) Original preference relation $\succ_1$

(b) Contracted preference relation $\succ_2$

Figure 1: Contraction of a finite preference relation

The examples above show that to discard a subset CON of a preference relation $\succ$, some preferences additional to CON may be discarded to make the resulting preference relation an SPO. A subset P^- of $\succ$ which containts CON and whose removal from $\succ$ preserves the SPO axioms of the modified preference relation is called a *full contractor of* $\succ$ *by* CON. Such a set P^- may be viewed as a union of the preferences CON to discard and *a set of reasons* of discarding CON. Ideally, if a user decides to discard preferences, she also provides all the reasons for such a change. In this case, the relation $(\succ - CON)$ is already a preference relation (i.e., SPO). However, in real life scenarios, it is hard to expect that users always provide complete information about the change they want to make. At the same time, the number of alternative full contractors P^- for a given $\succ$ and CON may be large or even infinite for infinite preference relations. As a result, there is often a need to learn from the user the reasons for discarding preferences. That may be done in a step-wise manner by exploring possible alternatives and using user feedback to select the correct ones.

We envision the following scenario here. To find a complete set of preferences she wants to discard, the user iteratively expresses the most obvious preferences CON that should be dropped from her preference relation $\succ$. After that, a possible set of reasons P^- for such a change is computed. To check if she is satisfied with the computed P^-, an impact of the performed change may be demonstrated to her (e.g., the result of the winnow operator over a certain data set). If the full contractor P^- does not represent the change she actually wanted to make, the user may undo the change and select another alternative or tune the contraction by elaborating it. One type of such elaboration is specifying a set of additional preferences to discard. Another type of elaboration which we propose in this paper is *preference protection*. Because the exact reasons for contracting CON are not known beforehand, some preferences which are important for the user may be contracted in an intermediate P^-. To avoid that, a user can impose a requirement of *protecting a set of preferences from removal*. The corresponding contraction operator is called here *preference-protecting contraction*. This iterative process stops when the user is satisfied with the computed full contractor.

An important property of the scenario above is that the set of reasons P^- which is computed as a result of the iterative process above has to be as small as possible. That is, preferences that the user does not want to discard and that are not needed to be removed to preserve the SPO properties of the modified preference relation should remain. To preserve the minimality of preference change, two approaches are possible.

In the first one, the full contractor computed in every step is *minimal*. The corresponding contraction operator here is called *minimal preference contraction*. It is guaranteed that the full contractor P^- computed in the last iteration (i.e., when P^- is satisfactory for the user) is minimal. Note that since there could be many possible minimal full contractors of a preference relation by a base contractor, *any* of them may be picked assuming that if the user is not completely satisfied with it, she will tune the contraction in the next iteration. In belief revision theory, the contraction operator with a similar semantics is called *maxichoice contraction* [Han98].

In the other variant, the full contractor P^- computed in every step is not necessary minimal. However, the user can make P^- smaller by specifying the preferences which should be protected from removal. The full contractor computed in the last step may be not minimal, but sufficiently small to meet the user expectations. We propose to construct P^- as the *union* of all minimal full contractors of the preference relation by CON. This contraction operator is called *meet contraction* if no preferences need to be preserved, and *preference-protecting meet contraction* if preference preservation is required. Similar operators in belief revision are *full meet contraction*, and *partial meet contraction* [Han98].

We note that the operations of preference contraction we propose in this paper should be understood in the context of the scenario discussed above. However, the details of the scenario are beyond the scope of this work. The main results of the paper are as follows. First, we present necessary and sufficient conditions for successful construction of the minimal preference contraction. Second, we propose two algorithms for minimal preference contraction: the first for finitely representable preference relations and the second for finite preference relations. Third, we show necessary and sufficient conditions for successful evaluation of the operator of preference-protecting contraction and propose an algorithm of computing the operator of preference-protecting minimal contraction. Fourth, we show how meet and meet preference-protecting contraction operators can be computed in the preference relation framework. Fifth, we show how to optimize preference query evaluation in the presence of contraction. Finally, we perform experimental evaluation of the proposed framework and present the results of the experiments. In the related work section, we show relationships of the current work with belief revision and other approaches of preferences change.

2. Basic Notions

The preference relation framework we use in the paper is a variation of the one proposed in [Cho03]. Let $\mathcal{A} = \{A_1, \ldots, A_m\}$ be a fixed set of attributes. Every attribute A_i is associated with a *domain* $\mathcal{D}_{A_i}$. We consider here two kinds of infinite domains: C (uninterpreted constants) and Q (rational numbers). Then a universe $\mathcal{U}$ of tuples is defined as

$$\mathcal{U} = \prod_{A \in \mathcal{A}} \mathcal{D}_{A_i}$$

We assume that two tuples o and o' are equal if and only if the values of their corresponding attributes are equal.

Definition 1. *A binary relation $\succ \subset \mathcal{U} \times \mathcal{U}$ is a* preference relation*, if it is a strict partial order (SPO) relation, i.e., transitive and irreflexive.*

Binary relations $R \subseteq \mathcal{U} \times \mathcal{U}$ considered in the paper are *finite* or *infinite*. Finite binary relations are represented as sets of pairs of tuples. The infinite binary relations we consider here are *finitely representable* as *formulas*. Given a binary relation R, its formula representation is denoted F_R. That is, $R(o, o')$ iff $F_R(o, o')$. A formula representation $F_\succ$ of a preference relation $\succ$ is called a *preference formula*.

We consider two kinds of atomic formulas here:

- *equality constraints*: $o.A_i = o'.A_i$, $o.A_i \neq o'.A_i$, $o.A_i = c$, or $o.A_i \neq c$, where o, o' are tuple variables, A_i is a C -attribute, and c is an uninterpreted constant;
- *rational-order constraints*: $o.A_i \theta o'.A_i$ or $o.A_i \theta c$, where $\theta \in \{=, \neq, <, >, \leq, \geq\}$, o, o' are tuple variables, A_i is a Q -attribute, and c is a rational number.

A preference formula whose all atomic formulas are equality (resp. rational-order) constraints will be called an *equality* (resp. *rational order*) preference formula. If both equality and rational order constraints are used in a formula, the formula will be called an *equality/rational order* formula or simply *ERO*-formula. Without loss of generality, we assume that all preference formulas are quantifier-free because ERO-formulas admit quantifier elimination. Examples of relations represented using ERO-formulas are $\succ$, $\succ_1$ and $\succ_2$ in Example 1.

We also use the representation of binary relations as *directed graphs*, both in the finite and the infinite case.

Definition 2. *Given a binary relation $R \subseteq \mathcal{U} \times \mathcal{U}$ and two tuples x and y such that xRy ($xy \in R$), xy* is an R-edge from x to y. *A* path in R *(or an* R-path*) from x to y is a sequence of R-edges such that the start node of the first edge is x, the end node of the last edge is y, and the end node of every edge (except the last one) is the start node of the next edge in the sequence. The sequence of nodes participating in an R-path is an* R-sequence*. The* length of an R-path *is the number of R-edges in the path. The* length of an R-sequence *is the number of nodes in it.*

An element of a preference relation is called *a preference*. We use the symbol $\succ$ with subscripts to refer to preference relations. We write $x \succeq y$ as a shorthand for $(x \succ y \vee x = y)$. We also say that x *is preferred to* y and y *is dominated by* x according to $\succ$ if $x \succ y$.

In this paper, we present several algorithms for finite relations. Such algorithms are implemented using the *relational algebra* operators: selection σ, projection π, join $\bowtie$, set difference $-$, and union $\cup$ [RG02]. Set difference and union in relational algebra have the same semantics as in the set theory. The semantics of the other operators are as follows:

- Selection $\sigma_F(R)$ picks from the relation R all the tuples for which the condition F holds. The condition F is a boolean expression involving comparisons between attribute names and constants.

- Projection $\pi_L(R)$ returns a relation which is obtained from the relation R by leaving in it only the columns listed in L and dropping the others.
- Join of two relations R and S

$$R \underset{R.X_1=S.Y_1,\ldots,R.X_n=S.Y_n}{\bowtie} S$$

computes a product of R and S, leaves only the tuples in which $R.X_1 = S.Y_1, \ldots, R.X_n = S.Y_n$, and drops the columns $S.Y_1, \ldots, S.Y_n$ from the resulting relation.

When we need more than one copy of a relation R in a relational algebra expression, we add subscripts to the relation name (e.g. R_1, R_2 etc).

3. Preference contraction

Preference contraction is an operation of discarding preferences. We assume that when a user intends to discard some preferences, he or she expresses the preferences to be discarded as a binary relation called a *base contractor*. The interpretation of each pair in a base contractor is that the first tuple should not be preferred to the second tuple. We require base contractor relations to be subsets of the preference relation to be contracted. Hence, a base contractor is irreflexive but not necessary transitive. Apart from that, we do not impose any other restrictions on the base contractors (e.g., they can be finite of infinite), unless stated otherwise. Throughout the paper, base contractors are typically referred to as CON.

Definition 3. *A binary relation P^- is* a full contractor of a preference relation $\succ$ by CON *if $CON \subseteq P^- \subseteq \succ$, and $(\succ - P^-)$ is a preference relation (i.e., an SPO). The relation $(\succ - P^-)$ is called the* contracted relation.

A relation P^- is a minimal full contractor of $\succ$ by CON *if P^- is a full contractor of $\succ$ by CON, and there is no other full contractor P' of $\succ$ by CON s.t. $P' \subset P^-$.*

Definition 4. *A preference relation is* minimally contracted *if it is contracted by a* minimal full contractor. Contraction *is an operation of constructing a full contractor.* Minimal contraction *is an operation of constructing a minimal full contractor.*

The notion of a minimal full contractor narrows the set of full contractors. However, as we illustrate in Example 3, a minimal full contractor is generally not unique for the given preference and base contractor relations. Moreover, the number of minimal full contractors for infinite preference relations can be infinite. Thus, minimal contraction differs from minimal preference revision [Cho07a] which is uniquely defined for given preference and revising relations.

Example 3. *Take the preference relation $\succ$ which is a total order of $\{x_1, \ldots, x_4\}$ (Figure 2). Let the base contractor relation CON be $\{x_1x_4\}$. Then the following sets are minimal full contractors of $\succ$ by CON: $P_1^- = \{x_1x_2, x_1x_3, x_1x_4\}$, $P_2^- = \{x_3x_4, x_2x_4, x_1x_4\}$, $P_3 = \{x_1x_2, x_3x_4, x_1x_4\}$, and $P_4^- = \{x_1x_3, x_2x_4, x_2x_3, x_1x_4\}$.*

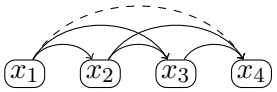


Figure 2: $\succ$ and CON

An important observation here is that that the contracted preference relation is defined as a subset of the original preference relation. We want to preserve the SPO properties – transitivity and irreflexivity – of preference relations. Since any subset of an irreflexive relation is also an irreflexive relation, no additional actions are needed to preserve irreflexivity during contraction. However, not every subset of a transitive relation is a transitive relation. We need to consider paths in the original preference relation which by transitivity may produce CON-edges which need to be discarded. We call such paths *CON-detours*.

Definition 5. *Let $\succ$ be a preference relation, and $P \subseteq \succ$. Then a $\succ$-path from x to y is a* P-detour *if $xy \in P$.*

First, let us consider the problem of finding any full contractor, not necessary minimal. As we showed above, a contracted preference relation cannot have any CON-detours. To achieve that, some additional edges of the preference relation have to be discarded. However, when we discard these edges, we have to make sure that there are no paths in the contracted preference relation which produce the removed edges. Hence, a necessary and sufficient condition for a subset of a preference relation *to be its full contractor* can be formulated in an intuitive way.

Lemma 1. *Given a preference relation (i.e., an SPO) $\succ$ and a full contractor CON, a relation $P^- \subseteq \succ$ is a full contractor of $\succ$ by CON if and only if $CON \subseteq P^-$, and for every $xy \in P^-$, $(\succ - P^-)$ contains no paths from x to y.*

Proof.
$\Leftarrow$ Prove that if for all $xy \in P^-$, $(\succ - P^-)$ contains no paths from x to y, then $(\succ - P^-)$ is an SPO. The irreflexivity of $(\succ - P^-)$ follows from the irreflexivity of $\succ$. Assume $(\succ - P^-)$ is not transitive, i.e., there are $xz, zy \in (\succ - P^-)$ but $xy \notin (\succ - P^-)$. If $xy \in P^-$ then the path xz, zy is not disconnected which contradicts the initial assumption. If $xy \notin P^-$, then the assumption of transitivity of $\succ$ is violated.
$\Rightarrow$ First, $CON \not\subseteq P^-$ implies that P^- is not a full contractor of $\succ$ by CON by definition. Second, the existence of a path from x to y in $(\succ - P^-)$ for $xy \in P^-$ implies that $(\succ - P^-)$ is not transitive, which violates the SPO properties. □

Now let us consider the property of minimality of full contractors. Let P^- be any minimal full contractor of a preference relation $\succ$ by a base contractor CON. Pick any edge xy of P^-. An important question which arises here is *why is xy a member of P^-?* The answer is obvious if xy is also a member of CON: every CON-edge has to be removed from the preference relation. However, what if xy is not a member of CON? To answer this question, let us introduce the notion of the *outer edge set* of an edge belonging to a full contractor relation.

Definition 6. *Let CON be a base contractor of a preference relation $\succ$, and P^- be a full contractor of $\succ$ by CON. Let $xy \in P^- - CON$, and*

$\Phi_0(xy) = \{xy\}$, *and*
$\Phi_i(xy) = \{u_iv_i \in P^- | \exists u_{i-1}v_{i-1} \in \Phi_{i-1}(xy)\ .\ u_i = u_{i-1} \wedge v_{i-1}v_i \in (\succ - P^-) \vee$
$v_{i-1} = v_i \wedge u_iu_{i-1} \in (\succ - P^-)\}$, *for* $i > 0$.

Then the outer edge set $\Phi(xy)$ for xy *is defined as*

$$\Phi(xy) = \bigcup_{i=0}^{\infty} \Phi_i(xy).$$

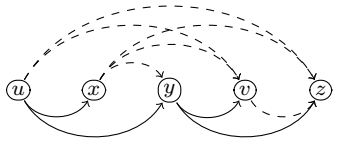


Figure 3: $\Phi(xy)$ for Example 4.

Intuitively, the outer edge set $\Phi(xy)$ of an edge $xy \in (P^- - CON)$ contains all the edges of a full contractor P^- which should be removed from P^- (i.e., added back to the preference relation $\succ$) to preserve the full contractor property of the result, should xy be removed from P^- (i.e., added back to the preference relation). The reasoning here is as follows. When for some i, $\Phi_i(xy)$ is removed from P^-, then $\Phi_{i+1}(xy)$ has to be also removed from P^-. Otherwise, for every edge in $\Phi_{i+1}(xy)$, there is a two-edge path in $\succ$ one of whose edges is in $\Phi_i(xy)$ while the other is not contracted. Hence, if the SPO properties of $(\succ - P^-)$ need to be preserved, removing xy from P^- requires recursively removing the entire $\Phi(xy)$ from P^-.

The next example illustrates the inductive construction of an outer edge set. Some properties of outer edge sets are shown in Lemma 2.

Example 4. *Let a preference relation $\succ$ be the set of all edges in Figure 3, and P^- be defined by the dashed edges. Let us construct $\Phi(xy)$ (assuming that xy is not an edge of the base contractor CON).*

- $\Phi_0(xy) = \{xy\}$;
- $\Phi_1(xy) = \{xv, xz\}$;
- $\Phi_2(xy) = \{uv, uz\}$;

Thus, $\Phi(xy) = \{xy, xv, xz, uv, uz\}$.

Lemma 2. *Let P^- be a full contractor of a preference relation $\succ$ by a base contractor CON. Then for every $xy \in (P^- - CON)$, $\Phi(xy)$ has the following properties:*

1. *for all* $uv \in \Phi(xy)$, $u \succeq x$ *and* $y \succeq v$;
2. *for all* $uv \in \Phi(xy)$, $ux, yv \notin P^-$;
3. *if* $(\Phi(xy) \cap CON) = \emptyset$, *then* $P' = (P^- - \Phi(xy))$ *is a full contractor of* $\succ$ *by* CON.

Proof. First, we prove that Properties 1 and 2 hold. We do it by induction on the index of $\Phi_i(xy)$ used to construct $\Phi(xy)$. For every $uv \in \Phi_0(xy)$, Properties 1 and 2 hold by the construction of Φ_0. Now let Properties 1 and 2 hold for $\Phi_n(xy)$, i.e.,

$$\forall u_n v_n \in \Phi_n(xy) \rightarrow u_n \succeq x \wedge y \succeq v_n \wedge u_n x, y v_n \notin P^- \tag{1}$$

Pick any $u_{n+1}v_{n+1} \in \Phi_{n+1}(xy)$. By construction of $\Phi_{n+1}(xy)$, we have

$$\begin{aligned} \exists u_n v_n \in \Phi_i(xy) \;.\; & u_{n+1} = u_n \wedge v_n \succ v_{n+1} \wedge v_n v_{n+1} \notin P^- \vee \\ & u_{n+1} \succ u_n \wedge v_n = v_{n+1} \wedge u_{n+1} u_n \notin P^- \end{aligned} \tag{2}$$

Note that $u_{n+1} \succeq x$ and $y \succeq v_{n+1}$ follows from (1), (2), and transitivity of $\succeq$. Similarly, $u_{n+1}x, yv_{n+1} \notin P^-$ is implied by (1), (2), and transitivity of $(\succ - P^-)$. Hence, Properties 1 and 2 hold for $\cup_{i=0}^{n} \Phi_i(xy)$ for any n.

Now we prove Property 3: $(\succ - P')$ is an SPO and $CON \subseteq P'$. The latter follows from $CON \subseteq P^-$ and $\Phi(xy) \cap CON = \emptyset$. Irreflexivity of $(\succ - P')$ follows from irreflexivity of $\succ$. Assume $(\succ - P')$ is not transitive, i.e., there are $uv \notin (\succ - P')$ and $uz, zv \in (\succ - P')$. Transitivity of $(\succ - P^-)$ implies that at least one of uz, zv is in $\Phi(xy)$. However, Property 1 implies that *exactly one* of uz, zv is in $\Phi(xy)$ and the other one is not in $\Phi(xy)$ and thus in $(\succ - P^-)$. However, $uz \in \Phi(xy)$ and $zv \in (\succ - P^-)$ imply $uv \in \Phi(xy)$, and thus $uv \in (\succ - (P^- - \Phi(xy))) = (\succ - P')$, i.e., we derive a contradiction. A similar contradiction is derived in the case $uz \in (\succ - P^-)$ and $zv \in \Phi(xy)$. Therefore, $(\succ - P')$ is an SPO and P' is a full contractor of $\succ$ by CON. □

Out of the three properties shown in Lemma 2, the last one is the most important. It says that if an edge xy of a full contractor is not needed to disconnect any CON-detours, then that edge may be dropped from the full contractor along with its entire outer edge set. A more general result which follows from Lemma 2 is formulated in the next theorem. It represents a necessary and sufficient condition for a full contractor to be *minimal*.

Theorem 1. (Full-contractor minimality test). *Let* P^- *be a full contractor of* $\succ$ *by* CON. *Then* P^- *is a* minimal *full contractor of* $\succ$ *by* CON *if and only if for every* $xy \in P^-$, *there is a* CON*-detour in* $\succ$ *in which* xy *is the only* P^-*-edge.*

Proof.
(⇐) The proof in this direction is straightforward. Assume that for every edge of the full contractor P^- there exists at least one CON-detour in which only that edge is in P^-. If P^- loses any its subset P containing that edge, then there will be a CON-detour in $\succ$ having no edges in $(P^- - P)$, and thus $(P^- - P)$ is not a full contractor of $\succ$ by CON by Lemma 1. Hence, P^- is a minimal full contractor.

$\boxed{\Rightarrow}$ Let P^- be a minimal full contractor. For the sake of contradiction, assume for some $xy \in P^-$, 1) there is no CON-detour which xy belongs to, or 2) any CON-detour xy belongs to has at least one more P^--edge. If 1) holds, then $\Phi(xy)$ has no edges in CON by construction. Thus, Lemma 2 implies that $(P^- - \Phi(xy))$ is a full contractor of $\succ$ by CON. Since $\Phi(xy)$ is not empty, we get that P^- is not a minimal full contractor which is a contradiction. If 2) holds, then we use the same argument as above and show that $\Phi(xy) \cap CON = \emptyset$. If $\Phi(xy) \cap CON$ is not empty (i.e., some $uv \in \Phi(xy) \cap CON$), then by Lemma 2,

$$u \succeq x \wedge x \succ y \wedge y \succeq v \wedge ux, yv \notin P^-,$$

and thus there is a CON-detour going from u to v in which xy is the only P^--edge. This contradicts the initial assumption. □

Note that using the definition of minimal full contractor to check the minimality of a full contractor P^- requires checking the full contractor properties of *all subsets* of P^-. In contrast, the minimality checking method shown in Theorem 1 requires checking properties of *distinct elements* of P^- with respect to its other members.

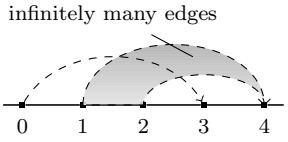
Sometimes a direct application of the minimality test from Theorem 1 is hard because it does not give any bound on the length of CON-detours. Hence, it is not clear how it can be represented as a finite formula. Fortunately, the transitivity of preference relations implies that the minimality condition from Theorem 1 can be stated in terms of paths of length at most three.

Corollary 1. *A full contractor P^- of $\succ$ by CON is* minimal *if and only if for every edge $xy \in P^-$, there is a CON-detour consisting of at most three edges among which only xy is in P^-.*

Proof.
$\boxed{\Leftarrow}$ Trivial.
$\boxed{\Rightarrow}$ For every $xy \in P^-$, pick any CON-detour T in which the only P^--edge is xy. If its length is less or equal to three, then the corollary holds. Otherwise, x is not the start node of T, or y is not the end node of T, or both. Let the start node u of T be different from x. Since the only common edge of T and P^- is xy, every edge in the path from u to x is an element of $(\succ - P^-)$. Transitivity of $(\succ - P^-)$ implies $ux \in (\succ - P^-)$. Similarly, $yv \in (\succ - P^-)$ for the end node of T if y is different from v. Hence, there is a CON-detour of length at most three in which xy is the only element of P^-. □

As a result, the following tests can be used to check the minimality of a full contractor P^-. In the finite case, P^- is minimal if the following relational algebra expression results in an empty set

$$\begin{aligned}
\mathrm{P} - [&\pi_{\mathrm{P_2.X,P_2.Y}}((\mathrm{R_1} - \mathrm{P_1}) \underset{\mathrm{R_1.Y=P_2.X}}{\bowtie} \mathrm{P_2} \underset{\mathrm{P_2.Y=R_3.X}}{\bowtie} (\mathrm{R_3} - \mathrm{P_3}) \underset{\mathrm{R_1.X}=C.X,\ \mathrm{R_3.Y=C.Y}}{\bowtie} \mathrm{C}) \ \cup \\
&\pi_{\mathrm{P_2.X,P_2.Y}}(\mathrm{P_2} \underset{\mathrm{P_2.Y=R_3.X}}{\bowtie} (\mathrm{R_3} - \mathrm{P_3}) \underset{\mathrm{P_2.X}=C.X,\ \mathrm{R_3.Y=C.Y}}{\bowtie} \mathrm{C}) \ \cup \\
&\pi_{\mathrm{P_2.X,P_2.Y}}((\mathrm{R_1} - \mathrm{P_1}) \underset{\mathrm{R_1.Y=P_2.X}}{\bowtie} \mathrm{P_2} \underset{\mathrm{R_1.X}=C.X,\ \mathrm{P_2.Y=C.Y}}{\bowtie} \mathrm{C}) \ \cup \mathrm{C}\ ],
\end{aligned}$$



(a) Infinite case, Example 5

R	X	Y
	u	x
	x	y
	y	v
	u	y
	x	v
	u	v

C	X	Y
	u	v

P	X	Y
	u	x
	y	v
	x	v
	u	v

D	X	Y
	u	x
	x	v

(b) Finite case, Example 6

Figure 4: Checking minimality of a full contractor

for the tables R, C and P with columns X and Y, storing $\succ$, CON, and P^- correspondingly. In the finitely representable case, P^- is minimal if the following formula is valid

$$\forall x, y\ (F_{P^-}(x,y) \Rightarrow F_{\succ}(x,y) \wedge \exists u, v\ .\ F_{CON}(u,v) \wedge (F_{\succ}(u,x) \vee u = x) \wedge (F_{\succ}(y,v) \vee y = v) \wedge \neg F_{P^-}(u,x) \wedge \neg F_{P^-}(y,v)).$$

Below we show examples of checking minimality of full contractors using Corollary 1. We note that when the relations are definable using ERO-formulas, checking minimality of a full contractor can be done by performing quantifier elimination on the above formula.

Example 5. *Let a preference relation $\succ$ be defined by the formula $F_{\succ}(o, o') \equiv o.d < o'.d$, where d is a Q -attribute. Let a base contractor CON of $\succ$ be defined by the formula*

$$F_{CON}(o, o') \equiv (1 \leq o.d \leq 2 \wedge o'.d = 4) \vee (o.d = 0 \wedge o'.d = 3)$$

(Figure 4(a)). Denote the relation represented by the first and second disjuncts of F_{CON} as CON_1 and CON_2 correspondingly. The relation P^- defined by F_{P^-} is a full contractor of $\succ$ by CON

$$F_{P^-}(o, o') \equiv (1 \leq o.d \leq 2 \wedge 2 < o'.d \leq 4) \vee (o.d = 0 \wedge 0 < o.d' \leq 3).$$

Similarly, denote the relations represented by the first and the second disjuncts of F_{P^-} as P_1^- and P_2^- correspondingly. We use Corollary 1 to check the minimality of P^-. By the corollary, we need to consider CON-detours of length at most three. Note that every P_1^--edge starts a one- or two-edge CON-detour with the corresponding CON_1-edge. Moreover, the second edge of all such two-edge detours is not contracted by P^-. Hence, the minimal full contractor test is satisfied for P_1^--edges. Now we consider P_2^--edges. All CON-detours which these edges belong to 1) correspond to CON_2-edges, and 2) are started by P_2^--edges. Hence, we need to consider only CON_2-detours of length at most two. When a P_2^--edge ends in o' with the value of d in $(0, 1)$ and $(2, 3]$, the second edge in the corresponding two-edge CON_2-detour is not contracted by P^-. However, when d is in $[1, 2]$, the second edge is already in P^-. Hence, P^- is not minimal by Corollary 1. To minimize it, we construct P^* by removing the edges from P^- which end in o' with d in $[1, 2]$

$$F_{P^*}(o, o') \equiv (1 \leq o.d \leq 2 \wedge 2 < o'.d \leq 4) \vee (o.d = 0 \wedge (0 < o.d' < 1 \vee 2 < o'.d \leq 3))$$

Example 6. *Take a preference represented by the table R, and a base contractor R represented by the table C (Figure 4(b)). Consider the table P representing a base contractor of R by C. Then the result of the relational algebra expression above evaluated for these tables is shown in the table D. Since it is not empty, the full contractor represented by P is not minimal. The minimality of P can be achieved by removing from it any (but only one) tuple in D.*

4. Construction of a minimal full contractor

In this section, we propose a method of computing a minimal full contractor. We use the idea shown in Example 3. Pick for instance the set P_1^-. That set was constructed as follows: we took the CON-edge x_1x_4 and put in P_1^- all the edges which start some path from x_1 to x_4. For the preference relation $\succ$ from Example 3, P_1^- turned out to be a minimal full contractor. As is it shown in the next lemma, the set consisting of all edges starting CON-detours is a full contractor by CON.

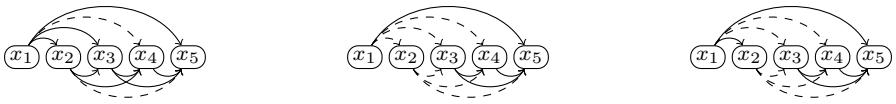
Lemma 3. *Let $\succ$ be a preference relation and CON be a base contractor relation of $\succ$. Then*

$$P^- := \{\ xy \mid \exists x'v \in CON\ .\ x' = x \wedge x' \succ y \wedge y \succeq v\}$$

is a full contractor of $\succ$ by CON.

Proof. By construction of P^-, $CON \subseteq P^-$. Lemma 1 implies $(\succ - P^-)$ is an SPO. Hence, $(\succ - P^-)$ is a full contractor of $\succ$ by CON. □

However, in the next example we show that such a full contractor is not always minimal. Recall that by Theorem 1, for every edge of a full contractor there should be a CON-detour which only shares that edge with the contractor. However, it may be the case that an edge starting a CON-detour does not have to be discarded because the CON-detour is already disconnected.



(a) Preference relation $\succ$ (b) Preference relation $(\succ - P^-)$ (c) Minimally contracted $\succ$

Figure 5: Preference contraction

Example 7. *Let a preference relation $\succ$ be a total order of $\{x_1, \ldots, x_5\}$ (Figure 5(a)). Let a base contractor CON be $\{x_1x_4, x_2x_5\}$. Let P^- be defined as in Lemma 3. That is $P^- = \{x_1x_2, x_1x_3, x_1x_4, x_2x_3, x_2x_4, x_2x_5\}$. Then $(\succ - P^-)$ is shown in Figure 5(b) as the set of solid edges. P^- is not minimal because $(P^- - \{x_1x_2\})$ (Figure 5(c)) is also a full contractor of $\succ$ by CON. In fact, $(P^- - \{x_1x_2\})$ is a* minimal *full contractor of $\succ$ by CON. As we can see, having the edge x_1x_2 in P^- is not necessary. First, it is not a CON-edge. Second, the edge x_2x_4 of the CON-detour $x_1 \succ x_2 \succ x_4$ is already in P^-.*

As we have shown in Example 7, a minimal full contractor can be constructed by including in it only the edges which start some CON-detour, if the detour is not already disconnected. Thus, before adding such an edge to a full contractor, we need to know if an edge *not starting* that detour is already in the full contractor. Here we propose the following idea of computing a minimal full contractor. Instead of contracting $\succ$ by CON at once, split CON into *strata* , and contract $\succ$ incrementally by the strata of CON. A stratum of CON consists of only those edges whose detours can be disconnected simultaneously in a minimal way. The method of splitting a full contractor into strata we propose to use is as follows.

Definition 7. *The* stratum index *of an edge $xy \in CON$ is* the maximum length of a $\succ$-path started by y and consisting of the end nodes of CON-edges. *A* stratum *is the set of all CON-edges with the same* stratum index *.*

This method of stratification has the following useful property. If a preference relation is contracted minimally by the strata with indices of up to n, then contracting that relation minimally by the stratum with the index $n+1$ minimally guarantees the minimality of the entire contraction.

Clearly, if a preference relation is infinite, a tuple can start $\succ$-paths of arbitrarily large lengths. Therefore, the stratum index of CON-edge may be undefined. We exclude such cases here, so we can assume that for each edge of CON relations, the stratum index is defined.

Definition 8. *Let CON be a base contractor of a preference relation $\succ$. Let $K_{CON} = \{y \mid \exists x \ . \ xy \in CON\}$, and $\succ_{CON} = \succ \cap K_{CON} \times K_{CON}$. Then CON is* stratifiable *iff for every $y \in K_{CON}$ there is an integer k such that all the paths started by y in $\succ_{CON}$ are of length at most k. CON is* finitely stratifiable *iff there is a constant k such that all paths in $\succ_{CON}$ are of length at most k.*

Definition 8 implies that for every edge of stratifiable CON, the stratum index is defined. Since the shortest path in $\succ_{CON}$ is of length 0, the least stratum index for stratifiable relations is 0. Instances of finitely stratifiable base contractors are shown in Example 1 ($k = 0$), Example 2 ($k = 1$), and Example 5 ($k = 1$). Below we present an approach of constructing a minimal full contractor for a *stratifiable* relation CON.

Theorem 2. (Minimal full contractor construction). *Let $\succ$ be a preference relation, and CON be a stratifiable base contractor of $\succ$. Let L_i be the set of the end nodes of all CON-edges of stratum i. Then P^-, defined as follows, is a* minimal full contractor *of $\succ$ by CON*

$$P^- = \bigcup_{i \in 0}^{\infty} E_i,$$

where

$$E_i = \{xy \mid \exists v \in L_i \ . \ xv \in CON \wedge x \succ y \wedge y \succeq v \wedge yv \notin (P^-_{i-1} \cup CON)\}$$

$$P^-_{-1} = \emptyset,$$

$$P^-_i = \bigcup_{j=0}^{i} E_i$$

Intuitively, the set E_i contains all the CON edges of stratum i along with the edges of $\succ$ which need to be discarded to contract the preference relation by that stratum. P^-_i is the union of all such sets up to stratum i.

Proof of Theorem 2. Every E_i containts the CON-edges of stratum i. Thus, P^- contains CON. Now we prove that $(\succ - P^-)$ is an SPO. Its irreflexivity follows from the irreflexivity of $\succ$. Transitivity is proved by induction on stratum index.

It is given that $\succ$ is transitive. Now assume $(\succ - P^-_n)$ is transitive. Prove that $(\succ - P^-_{n+1}) = (\succ - P^-_n - E_{n+1})$ is transitive. For the sake of contradiction, assume

$$\exists x, y, z \;.\; xy \notin (\succ - P^-_{n+1}) \wedge xz, zy \in (\succ - P^-_{n+1}) \tag{1}$$

which implies

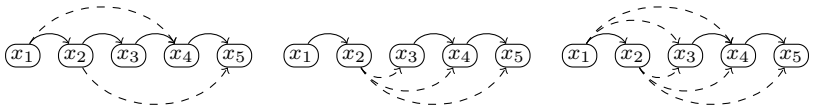
$$xz, zy \notin E_{n+1} \cup P^-_n \tag{2}$$

Transitivity of $(\succ - P^-_n)$ and (1) imply $xy \in (\succ - P^-_n)$ and thus $xy \in E_{n+1}$. Hence,

$$\exists v \in L_n \;.\; xv \in CON \wedge x \succ y \wedge y \succeq v \wedge yv \notin (P^-_n \cup CON) \tag{3}$$

According to (3), $y \succeq v$. If $y = v$, then (2) and (3) imply $xz \in E_{n+1}$ which is a contradiction. If $y \succ v$, then $xz \notin E_{n+1}$ implies $zv \in P^-_n \cup CON$ by the construction of E_{n+1}. Note that $zv \in CON$ implies zv is a CON-edge of stratum index $n+1$ and thus either $zy \in E_{n+1}$ or $yv \in P^-_n \cup CON$, which contradicts (2) and (3). If $zv \in P^-_n$, then $zy, yv \notin P^-_n$ implies intransitivity of $(\succ - P^-_n)$, which contradicts the inductive assumption. Thus, P^-_{n+1} is a full contractor of $\succ$ by CON by induction. Now assume that $(\succ - P^-)$ is not transitive. Violation of transitivity means that there is an edge $xy \in P^-$ such that there exists a path from x to y none of whose edges is P^- (Lemma 1). Since xy must be in P^-_n for some n, that implies intransitivity of $(\succ - P^-_n)$, which is a contradiction. Thus P^- is a full contractor of $\succ$ by CON.

Now we prove that P^- is a *minimal* full contractor. If it is not, then by Theorem 1, there is $xy \in P^-$ for which there is no CON-detour which shares with P^- only the edge xy. Note that $xy \in P^-$ implies $xy \in E_n$ for some n. By definition of E_n, there is a CON-detour $x \succ y \succeq v$ which shares with P^-_n only xy. Since all CON-detours which xy belongs to have other P^--edges, $yv \in P^-$. Since $yv \notin P^-_n$, there must exist $k > n$ such that $yv \in E_k$. However, that is impossible by construction: every CON-detour which may be started by yv must have the stratum index not greater than n. □



(a) $\succ$ and CON (b) P_0^- (c) P_1^-

Figure 6: Using Theorem 2 to compute a minimal full contractor

Example 8. *Let a preference relation $\succ$ be a total order of $\{x_1, \ldots, x_5\}$ (Figure 6(a), the transitive edges are omitted for clarity). Let a base contractor CON be $\{x_1x_4, x_2x_5\}$. We use Theorem 2 to construct a minimal full contractor of $\succ$ by CON. The relation CON has two strata: $L_0 = \{x_2x_5\}$, $L_1 = \{x_1x_4\}$. Then $E_0 = \{x_2x_3, x_2x_4, x_2x_5\}$, $P_0^- = E_0$, $E_1 = \{x_1x_3, x_1x_4\}$, $P_1^- = E_0 \cup E_1$, and a minimal full contractor of $\succ$ by CON is $P^- = P_1^-$.*

It is easy to observe that the full contractor P^- constructed in Theorem 2 has the property that its every edge *starts* at least one CON-detour in which xy is the only P^--edge. Full contractors which have this property are called *prefix*. Prefix full contractors are minimal by Theorem 1. It turns out that a prefix full contractor is unique for a given preference relation and a given base contractor.

Proposition 1. *Given a preference relation $\succ$ and a base contractor CON stratifiable, there exists a unique prefix full contractor P^- of $\succ$ by CON.*

Proof. The existence of a prefix full contractor follows from Theorem 2. The fact that every prefix full contractor is equal to P^- constructed by Theorem 2 can be proved by induction in CON stratum index. Namely, we show that for every n, P_n^- is contained in any prefix full contractor of $\succ$ by CON. Clearly, the set E_0 contracting $\succ$ by the 0^{th} stratum of CON has to be in any prefix full contractor. Assume every edge in P_n^- is in any prefix full contractor of $\succ$ by CON. If an edge $xy \in E_{n+1} - CON$, then there is a CON-detour $x \succ y \succ v$ in which xy is the only P^--edge (i.e., $yv \notin P^-$). Hence if xy is not in some prefix full contractor P', then yv has to be in P' by Lemma 1. However, $P_n^- \subset P'$ is enough to disconnect every CON-detour with index up to n, and yv can only start a CON-detour with the stratum index up to n. Hence P' is not a minimal full contractor and P^- is a unique prefix full contractor. □

5. Contraction by finitely stratifiable relations

In this section, we consider practical issues of computing minimal full contractors. In particular, we show how the method of constructing a prefix full contractor we have proposed in Theorem 2 can be adopted to various classes of preference and base contractor relations. Note that the definition of the minimal full contractor in Theorem 2 is recursive. Namely, to find the edges we need to discard for contracting the preference relation by the stratum $n+1$ of CON, we need to know which edges to discard for contracting it by all the previous strata. It means that for base contractor relations which are not finitely stratifiable (i.e., CON has infinite number of strata), the corresponding computation will never terminate.

Now assume that CON is a finitely stratifiable relation. First we note that any base contractor of a finite preference relation is finitely stratifiable: all paths in such preference

relations are not longer than the size of the relation, and base contractors are required to be subsets of the preference relations. At the same time, if CON is a base contractor of an infinite preference relation, then the finite stratifiability property of CON does not imply the finiteness of CON. In particular, it may be the case that the *length* of all paths in $\succ_{CON}$ is bounded, but the *number* of paths is infinite. This fact is illustrated in the next example.

Example 9. *Let a preference relation $\succ$ be defined as $o \succ o' \equiv o.price < o'.price$. Let every tuple have two Q -attributes: price and year. Let also the base contractor relations CON_1 and CON_2 be defined as*

$$CON_1(o, o') \equiv o.price < 1 \wedge (o'.price = 2 \vee o'.price = 3),$$
$$CON_2(o, o') \equiv o.price < 1 \wedge o'.price \geq 2.$$

then

$$K_{CON_1} \equiv \{o \mid o.price = 2 \vee o.price = 3\}$$
$$K_{CON_2} \equiv \{o \mid o.price \geq 2\}$$

and

$$o \succ_{CON_1} o' \equiv o.price = 2 \wedge o'.price = 3$$
$$o \succ_{CON_2} o' \equiv o.price \geq 2 \wedge o'.price > 2 \wedge o.price < o'.price$$

Then CON_1 is finitely stratifiable *since despite the fact that the number of edges in $\succ_{CON_1}$ and in CON_1 is infinite (due to the infiniteness of the domain of year), the length of the longest path in $\succ_{CON_1}$ equals to* 1*. Such paths are started by tuples with the value of price equal to* 2 *and ended by tuples with price equal to* 3*. At the same time, CON_2 is not finitely stratifiable since price is a Q -attribute and thus there is no constant bounding the length of all paths in $\succ_{CON_2}$.*

Below we consider the cases of finite and finitely representable finitely stratifiable base contractors separately.

5.1. Computing prefix full contractor: finitely representable relations

Here we assume that the relations CON and $\succ$ are represented by finite ERO-formulas F_{CON} and $F_{\succ}$. We aim to construct a finite ERO-formula F_{P^-} which represents a prefix full contractor of $\succ$ by CON. The function `minContr(`$F_{\succ}$`,` F_{CON}`)` shown below exploits the method of constructing prefix full contractors from Theorem 2 adopted to formula representations of relations. All the intermediate variables used in the algorithm store formulas. Hence, for example, any expression in the form $''F(x, y) := \ldots''$ means that the formula-variable F is assigned the formula written in the right-hand side, which has two free tuple variables x and y. The operator QE used in the algorithm computes a quantifier-free formula equivalent to its argument formula. For ERO-formulas, the operator QE runs in time

polynomial in the size of its argument formula (if the number of attributes in $\mathcal{A}$ is fixed), and exponential in the number of attributes in $\mathcal{A}$.

To compute formulas representing different strata of CON, `getStratum` (Algorithm 2) is used. It takes three parameters: the formula $F_{\succ_{CON}}$ representing the relation $\succ_{CON}$, the formula $F_{K_{CON}}$ representing the set of the end nodes of CON-edges, and the stratum index i. It returns a formula which represents the set of the end nodes of CON-edges of stratum i , or `undefined` if the corresponding set is empty. That formula is computed according to the definition of a stratum.

Algorithm 1 `minContr`$(F_{\succ},\ F_{CON})$

$i = 0$
$F_{P^-_{-1}}(x, y) := false$
$F_{K_{CON}}(y) := QE(\exists x\ .\ F_{CON}(x, y))$
$F_{\succ_{CON}}(x, y) := F_{CON}(x, y) \wedge F_{K_{CON}}(x) \wedge F_{K_{CON}}(y)$
$F_{L_i}(y) :=$ `getStratum`$(F_{\succ_{CON}}, F_{K_{CON}}, i)$
while F_{L_i} is defined **do**
    $F_{E_i}(x, y) := QE(\exists v\ .\ F_{L_i}(v) \wedge F_{CON}(x, v) \wedge F_{\succ}(x, y) \wedge (y = v \vee F_{\succ}(y, v) \wedge \neg(F_{P^-_{i-1}}(y, v) \vee F_{CON}(y, v))))$
    $F_{P^-_i}(x, y) := F_{P^-_{i-1}}(x, y) \vee F_{E_i}(x, y)$
    $i := i + 1$;
    $F_{L_i}(y) :=$ `getStratum`$(F_{\succ_{CON}}, F_{K_{CON}}, i)$
end while
return P^-_i

Algorithm 2 `getStratum`$(F_{\succ_{CON}},\ F_{K_{CON}},\ i)$

Require: $i \geq 0$
if i $= 0$ **then**
    $F_{L_i}(y) := QE(\ F_{K_{CON}}(y) \wedge \neg\exists x_1(F_{\succ_{CON}}(y, x_1)))$
else
    $F_{L_i}(y) := QE(\quad \exists x_1, \ldots, x_i \quad .\ F_{\succ_{CON}}(y, x_1) \wedge F_{\succ_{CON}}(x_1, x_2) \wedge \ldots \wedge F_{\succ_{CON}}(x_{i-1}, x_i)) \wedge \neg\exists x_1, \ldots, x_{i+1}\ .\ F_{\succ_{CON}}(y, x_1) \wedge F_{\succ_{CON}}(x_1, x_2) \wedge \ldots \wedge F_{\succ_{CON}}(x_i, x_{i+1})))$
end if
if $\exists y\ .\ F_{L_i}(y)$ **then**
    return F_{L_i}
else
    return `undefined`
end if

Proposition 2. *Let CON be a finitely stratifiable base contractor of a preference relation $\succ$. Then Algorithm 1 terminates and computes a prefix full contractor of $\succ$ by CON.*

Proposition 2 holds because Algorithm 1 uses the construction from Theorem 2. Below we show an example of computing a prefix full contractor for a finitely representable preference relation.

Example 10. *Let a preference relation $\succ$ be defined by the following formula*

$$F_{\succ}(o, o') \equiv o.m = BMW \wedge o'.m = VW \vee o.m = o'.m \wedge o.price < o'.price$$

and a base contractor CON be defined by

$$F_{CON}(o, o') \equiv o.m = o'.m \wedge ((11000 \leq o.price \leq 13000 \wedge o'.price = 15000) \vee (10000 \leq o.price \leq 12000 \wedge o'.price = 14000))$$

where m is a C -attribute and price is a Q -attribute. Then $F_{K_{CON}}(o) \equiv o.price = 14000 \vee o.price = 15000$ and $F_{\succ_{CON}}(o, o') \equiv F_{\succ}(o, o') \wedge F_{K_{CON}}(o) \wedge F_{K_{CON}}(o')$. The end nodes of the CON strata are defined by the following formulas:

$$\begin{aligned} F_{L_0}(o) \equiv & o.price = 15000 \wedge o.m \neq BMW \\ F_{L_1}(o) \equiv & o.price = 15000 \wedge o.m = BMW \vee o.price = 14000 \wedge o.m \neq BMW \\ F_{L_2}(o) \equiv & o.price = 14000 \wedge o.m = BMW. \end{aligned}$$

The relations contracting all CON strata are defined by the following formulas

$$\begin{aligned} F_{E_0}(o, o') \equiv & o.m = o'.m \neq BMW \wedge 11000 \leq o.price \leq 13000 \wedge 13000 < o'.price \leq 15000 \\ F_{E_1}(o, o') \equiv & o.m = o'.m = BMW \wedge 11000 \leq o.price \leq 13000 \wedge 13000 < o'.price \leq 15000 \vee \\ & o.m = o'.m \neq BMW \wedge 10000 \leq o.price < 11000 \wedge 13000 < o'.price \leq 14000 \\ F_{E_2}(o, o') \equiv & o.m = o'.m = BMW \wedge 10000 \leq o.price \leq 11000 \wedge 13000 < o'.price \leq 14000 \end{aligned}$$

Finally, a full contractor P^- of $\succ$ by CON is defined by

$$F_{P^-}(o, o') \equiv o.m = o'.m \wedge (11000 \leq o.price \leq 13000 \wedge 13000 < o'.price \leq 15000 \vee 10000 \leq o.price < 11000 \wedge 13000 < o'.price \leq 14000)$$

The finite stratifiability property of CON is crucial for the termination of the algorithm: the algorithm does not terminate for relations not finitely stratifiable. Hence, given a base contractor relation, it is useful to know if it is finitely stratifiable or not. Let us consider the formula $F_{\succ_{CON}}$. Without loss of generality, we assume it is represented in DNF. By definition, CON is a finitely stratifiable relation if and only if there is a constant k such that all $\succ_{CON}$ paths are of length at most k. In the next theorem, we show that this property can be checked by a single evaluation of the quantifier elimination operator.

Theorem 3. (Checking finite stratifiability property). *Let F_R be an ERO-formula, representing an SPO relation R, in the following form*

$$F_R(o, o') = F_{R_1}(o, o') \vee \ldots \vee F_{R_l}(o, o'),$$

where F_{R_i} is a conjunction of atomic formulas. Then checking if there is a constant k such that the length of all R-paths is at most k can be done by a single evaluation of QE over a formula of size linear in $|F_R|$.

In Theorem 3, we assume that each atomic formula using the operators $\leq, \geq$ is transformed to disjunction of two formulas: one which uses the strict comparison operator and the other using the equality operator. The proof of Theorem 3 and the details of the corresponding finite stratifiability property test are provided in Appendix A.

5.2. Computing prefix full contractor: finite relations

In this section, we consider finite relations $\succ$ and CON. We assume that the relations are stored in separate tables: a preference relation table R and a base contractor table C, each having two columns X and Y. Every tuple in a table corresponds to an element of the corresponding binary relation. Hence, R has to be an SPO and $C \subseteq R$. Here we present an algorithm of computing a prefix full contractor of a preference relation $\succ$ by CON represented by such tables. Essentially, the algorithm is an adaptation of Theorem 2.

The function `minContrFinite` takes two arguments: R and C. The function is implemented in terms of relational algebra operators. First, it constructs two tables: EC storing the end nodes of all C-edges, and RC storing a restriction of the original preference relation R to EC. These two tables are needed for obtaining the strata of C. After that, the function picks all strata of C one by one and contracts the original preference relation by each stratum in turn, as shown in Theorem 2.

The extraction of the strata of CON in the order of the stratum index is performed as follows. It is clear that the nodes ending CON-edges of stratum 0 do not start any edge in RC. The set E computed in line 8 is a difference of the set EC of the nodes ending C-edges and the nodes starting some edges in RC. Hence, E stores all the nodes ending C-edges of stratum 0. To get the end nodes of the next stratum of C, we need remove all the edges from RC which end in members of E, and remove E from EC. After the stratum with the highest index is obtained, the relation EC becomes empty.

Proposition 3. *Algorithm 3 computes a prefix full contractor of R by C. Its running time is $\mathcal{O}(|C|^2 \cdot |R| \cdot log|R|)$.*

Proposition 3 holds because Algorithm 3 uses the construction from Theorem 2. The stated running time may be obtained by applying some simple optimizations: (i) sorting EC after constructing it (line 3), (ii) sorting on X, Y the table R and the table RC right after its construction (line 5), (iii) keeping these relations sorted after every change. In addition to that, we store the relation P containing the intermediate full contractor edges as a copy of R, in which the edges which belong to the prefix full contractor are marked. By doing so, P is maintained in the sorted state throughout the algorithm.

Algorithm 3 `minContrFinite`(R, C)

Require: R is transitive, $C \subseteq R$

1: $P \leftarrow C$
2: /* Get the end nodes of all C-edges */
3: $EC \leftarrow \pi_Y(C)$
4: /* RC is related to R as $\succ_{CON}$ to $\succ$ in Definition 8 */
5: $RC \leftarrow \pi_{R.X,\, R.Y}\,(EC_1 \underset{EC_1.Y=R.X}{\bowtie} R \underset{EC_2.Y=R.Y}{\bowtie} EC_2)$
6: **while** EC not empty **do**
7: /* Get the end nodes of the next stratum C-edges */
8: $E \leftarrow EC - \pi_X(RC)$
9: /* Prepare EC and RC for the next iteration */
10: $EC \leftarrow EC - E$
11: $RC \leftarrow RC - RC \underset{RC.Y=E.Y}{\bowtie} E$
12: /* Add to P the R-edges contracting the current stratum of C*/
13: $P \leftarrow P \cup \pi_{R_1.X,\, R_1.Y}\,(R_1 \underset{R_1.Y\,=\,R_2.X}{\bowtie} (R_2 - P) \underset{R_1.X\,=\,C.X,\, R_2.Y\,=\,C.Y}{\bowtie} (C \underset{C.Y\,=\,E.Y}{\bowtie} E))$
14: **end while**
15: **return** P

6. Preference-protecting contraction

Consider the operation of minimal preference contraction described above. In order to contract a preference relation, a user has to specify a base contractor CON. The main criteria we use to define a contracted preference relation is *minimality of preference change*. However, a minimal full contractor P^- may contain additional preferences which are not in CON. So far, we have not paid attention to the contents of P^-, assuming that any minimal full contractor is equally good for a user. However, this may not be the case in real life. Assume that an original preference relation $\succ$ is combined from two preference relations $\succ = \succ_{old} \cup \succ_{recent}$, where $\succ_{old}$ describes user preferences introduced by the user a long time ago, and $\succ_{recent}$ describes more recent preferences. Now assume that the user wants to contract $\succ$ by CON, at least two minimal full contractors are possible: P_1^- which consists of CON and some preferences of $\succ_{old}$, and P_2^- consisting of CON and some preferences of $\succ_{recent}$. Since $\succ_{recent}$ has been introduced recently, discarding members of $\succ_{old}$ may be more reasonable then members of $\succ_{recent}$. Hence, sometimes there is a need to compute full contractors which *protect* some existing preferences from removal.

Here we propose an operator of *preference-protecting* contraction. In addition to a base contractor CON, a subset P^+ of the original preference relation to be *protected from removal* in the contracted preference relation may also be specified. Such a relation is complementary with respect to the base contractor: the relation CON defines the preferences to discard, whereas the relation P^+ defines the preferences to protect.

Definition 9. *Let $\succ$ be a preference relation and CON be a base contractor of $\succ$. Let a relation P^+ be such that $P^+ \subseteq \succ$. A full contractor P^- of $\succ$ by CON such that $P^+ \cap P^- = \emptyset$*

is called a P^+-protecting full contractor of $\succ$ by CON. *A minimal full contractor* P^- *of* $\succ$ *by* CON *such that* $P^+ \cap P^- = \emptyset$ *is called a* P^+-protecting minimal full contractor of $\succ$ by CON.

Given any full contractor P^- of $\succ$ by CON, by Lemma 1, P^- must contain at least one edge from every CON-detour. Thus, if P^+ contains an entire CON-detour, protecting P^+ while contracting $\succ$ by CON is not possible.

Theorem 4. *Let* CON *be a stratifiable base contractor relation of a preference relation* $\succ$ *such that* $P^+ \subset \succ$. *There exists a minimal full contractor of* $\succ$ *by* CON *that protects* P^+ *if and only if* $P^+_{TC} \cap CON = \emptyset$, *where* P^+_{TC} *is the transitive closure of* P^+.

As we noted, the necessary condition of the theorem above follows from Lemma 1. The sufficient condition follows from Theorem 5 we prove further.

A naive way of computing a preference-protecting minimal full contractor is by finding a minimal full contractor P^- of $(\succ - P^+)$ and then adding P^+ to P^-. However, $(\succ - P^+)$ is not an SPO in general, thus obtaining SPO of $\succ - (P^- \cup P^+)$ becomes problematic.

The solution we propose here uses the following idea. First, we find a base contractor CON' such that minimal contraction of $\succ$ by CON' is equivalent to minimal contraction of $\succ$ by CON with protected P^+. After that, we compute a minimal full contractor of $\succ$ by CON' using Theorem 2.

Recall that minimal full contractors constructed in Theorem 2 are *prefix*, i.e., every edge xy in such a full contractor starts some CON-detour in which xy is the only edge of the contractor. Thus, if no member of P^+ starts a CON-detour in $\succ$, then the minimal full contractor and P^+ have no common edges. Otherwise assume that an edge $xy \in P^+$ starts a CON-detour in $\succ$. By Lemma 1, any P^+-protecting full contractor P^- has to contain an edge different from xy which belongs to CON-detours started by xy. Moreover, for CON-detours of length two started by xy, P^- has to contain the *edges ending those* CON*-detours*. Such a set of edges is defined as follows:

$$Q = \{xy \mid \exists u : u \succ x \succ y \wedge uy \in CON \wedge ux \in P^+\}.$$

It turns out that the set Q is not only contained in any P^+-protecting full contractor, but it can also be used to construct a P^+-protecting minimal full contractor as shown in the next theorem.

Theorem 5. *Let* $\succ$ *be a preference relation, and* CON *be a stratifiable base contractor of* $\succ$. *Let also* P^+ *be a transitive relation such that* $P^+ \subseteq \succ$ *and* $P^+ \cap CON = \emptyset$. *Then the prefix full contractor of* $\succ$ *by* $CON \cup Q$ *is a* P^+*-protecting minimal full contractor of* $\succ$ *by* CON.

Proof. Let P^- be a prefix full contractor of $\succ$ by $CON' = CON \cup Q$. We prove that $P^- \cap P^+ = \emptyset$, i.e., P^- protects P^+. For the sake of contradiction, assume there is $xy \in P^+ \cap P^-$. We show that this contradicts the prefix property of P^-. Since P^- is a prefix full

(a) P^+-protection: case 1 (b) P^+-protection: case 2

Figure 7: Proof of Theorem 5

contractor, there is a CON'-detour from x to some v in $\succ$, started by xy and having only the edge xy in P^-. We have two choices: either it is a CON-detour or a Q-detour. Consider the first case. Clearly, $y \neq v$, otherwise $P^+ \cap CON \neq \emptyset$. Thus, $xv \in CON$ and $x \succ y \succ v$ (Figure 7(a)). $yv \in Q$ follows from $xy \in P^+$, $xv \in CON$ and the construction of Q. Note that every path from y to v in $\succ$ contains a P^--edge because P^- is a full contractor of $\succ$ by $CON \cup Q$. That implies that no CON-detour from x to v started by xy has only xy in P^- which contradicts the initial assumption.

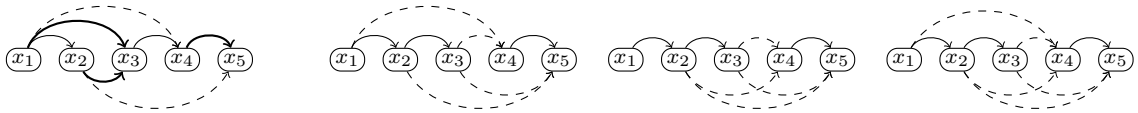
Consider the second case, i.e., there is a Q-detour from x to some v started by xy and having only the edge xy in P^-. Since $xv \in Q$, there is $uv \in CON$ such that $ux \in P^+$ (Figure 7(b)). $ux, xy \in P^+$ imply $uy \in P^+$ by transitivity of P^+. $uy \in P^+$ and $uv \in CON$ imply $yv \in Q$. That along with the fact that P^- is a full contractor of $\succ$ by $CON \cup Q$ implies that every path in $\succ$ from y to v contains a P^--edge. Hence, there is no Q-detour from x to v started by xy and having only xy in P^-. That contradicts the initial assumption about xy.

Now we prove that P^- is a minimal full contractor of $\succ$ by CON. The fact that it is a full contractor of $\succ$ by CON follows from the fact that it is a full contractor of $\succ$ by a superset CON' of CON. We prove now its minimality. Since P^- is a prefix full contractor of $\succ$ by CON', for every $xy \in P^-$, there is $xv \in CON'$ such that there is a corresponding detour T in which xy is the only P^--edge. If it is a CON-detour, then xy satisfies the minimality condition from Theorem 1. If it is a Q-detour, then there is a CON-edge uv such that $ux \in P^+$. We showed above that P^- protects P^+. Hence, the CON-detour obtained by joining the edge ux and T has only xy in P^-. Therefore, P^- is a minimal full contractor of $\succ$ by CON. □

Note that the sets of the end nodes of $(CON \cup Q)$-edges and the end nodes of CON-edges coincide by the construction of Q. Therefore, $(CON \cup Q)$ is stratifiable or finitely stratifiable if and only if CON is stratifiable or finitely stratifiable, correspondingly. Hence, if CON is a finitely stratifiable relation with respect to $\succ$, Algorithms 1 and 3 can be used to compute a preference-protecting minimal full contractor of $\succ$ by CON. If the relations $\succ$ and CON are finite, then Q can be constructed in polynomial time in the size of $\succ$ and CON by a relational algebra expression constructed from its definition. If the relations are finitely representable, then Q may be computed using the quantifier elimination operator QE.

For Theorem 5 to apply, the relation P^+ has to be transitive. Non-transitivity of P^+ implies that there are two edges $xy, yz \in P^+$ which should be protected while transitive

edge xz is not critical. However, a relation obtained as a result of preference-protecting contraction is a preference relation (i.e., SPO). Hence, the edge xz will also be protected in the resulting preference relation. This fact implies that protecting any relation is equivalent to protecting its minimal transitive extension: its transitive closure. Therefore, if P^+ is not transitive, one needs to compute its transitive closure to use Theorem 5. For finite relations, transitive closure can be computed in polynomial time [CLRS01]. For finitely representable relations, *Constraint Datalog* [KKR95] can be used to compute transitive closure.



(a) $\succ$, CON, and P^+ (b) $\succ$ and CON' (c) P_0^- (d) P_1^-

Figure 8: Using Theorem 5 to compute a preference-protecting minimal full contractor

Another important observation here is that the P^+-protecting minimal full contractor of $\succ$ by CON computed according to Theorem 5 is not necessary a *prefix* full contractor of $\succ$ by CON. This fact is illustrated in the following example.

Example 11. *Let a preference relation $\succ$ be a total order of $\{x_1, \ldots, x_5\}$ (Figure 8(a), the transitive edges are omitted for clarity). Let a base contractor CON be $\{x_1x_4, x_2x_5\}$, and $P^+ = \{x_1x_3, x_2x_3, x_4x_5\}$.*

The existence of a minimal P^+-protecting full contractor of $\succ$ by CON follows from Theorem 4. We use Theorem 5 to construct it. The set Q is equal to $\{x_3x_4, x_3x_5\}$ and $CON' = \{x_1x_4, x_2x_5, x_3x_4, x_3x_5\}$. We construct a prefix full contractor of $\succ$ by CON'. The relation CON' has two strata: $L_0 = \{x_2x_5, x_3x_5\}$, $L_1 = \{x_1x_4, x_3x_4\}$. Then $E_0 = \{x_2x_5, x_3x_5, x_2x_4, x_3x_4\}$, $P_0^- = E_0$, $E_1 = \{x_1x_4, x_3x_4\}$, $P_1^- = E_0 \cup E_1$, and $P^- = P_1^-$. By Theorem 5, P^- is a P^+-protecting minimal full contractor of $\succ$ by CON. However, P^- is not a prefix full contractor of $\succ$ by CON, because the edges x_3x_4, x_3x_5 do not start any CON-detour.

7. Meet preference contraction

In this section, we consider the operation of *meet preference contraction*. In contrast to the preceding sections, where the main focus was the minimality of preference relation change, the contraction operation considered here changes a preference relation not necessarily in a minimal way. A full meet contractor of a preference relation is semantically a *union* of all minimal sets of reasons of discarding a given set preferences. When a certain set of preferences is required to be protected while contracting a preference relation, the operation of *preference-protecting meet contraction* may be used.

(a) $\succ$ and CON_1 (b) $\succ$ and CON_2

Figure 9: Example 12

Definition 10. *Let $\succ$ be a preference relation, CON a base contractor of $\succ$, and $P^+ \subseteq \succ$. The relation P^m is a* full meet contractor *of $\succ$ by CON iff*

$$P^m = \bigcup_{P^- \in \mathcal{P}^m} P^-,$$

for the set $\mathcal{P}^m$ of all minimal full contractors of $\succ$ of CON. The relation $P^m_{P^+}$ is a full P^+-protecting meet contractor of $\succ$ by CON *iff*

$$P^m_{P^+} = \bigcup_{P^- \in \mathcal{P}^m_{P^+}} P^-,$$

for the set $\mathcal{P}^m_{P^+}$ of all P^+-protecting minimal full contractors of $\succ$ of CON.

Note that the relations $(\succ - P^m)$ and $(\succ - P^m_{P^+})$ can be represented as intersections of preference (i.e., SPO) relations and thus are also preference (i.e., SPO) relations. Let us first consider the problem of constructing full meet contractors.

By the definition above, an edge xy is in the full meet contractor of a preference relation $\succ$ by CON if there is a minimal full contractor of $\succ$ by CON which contains xy. Theorem 1 implies that if there is no CON-detour in $\succ$ containing xy, then xy is not in the corresponding full meet contractor. However, the fact that xy belongs to a CON-detour is not a sufficient condition for xy to be in the corresponding full meet contractor.

Example 12. *Let a preference relation $\succ$ be a total order of $\{u, x, y, v\}$. Let also $CON_1 = \{uv\}$ (Figure 9(a)) and $CON_2 = \{uv, yv\}$ (Figure 9(b)). There is only one CON_1- and CON_2-detour containing xy: $u \succ x \succ y \succ v$. There is also a minimal full contractor of $\succ$ by CON_1 which contains xy: $P_1^- = \{uy, xv, xy, uv\}$. However, there is no minimal full contractor of $\succ$ by CON_2 which contains xy because the edge yv of the CON_2-detour $u \succ x \succ y \succ v$ is in CON_2.*

In Theorem 6, we show how full meet contractors can be constructed in the case of *finitely stratifiable base contractors* . According to that theorem, a $\succ$-edge xy is in the full meet contractor of $\succ$ by CON if and only if there is a full contractor P^- of $\succ$ by CON such that xy is the only P^--edge in some CON-detour. We use Theorem 4 to show that there is a minimal full contractor of $\succ$ by CON which contains xy while the other edges of the detour are protected.

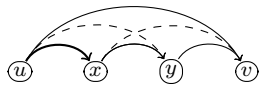


Figure 10: $\succ$, CON, and P^+ from Example 13

Theorem 6. *Let CON be a finitely stratifiable base contractor of a preference relation $\succ$. Then the full meet contractor of $\succ$ by CON is*

$$P^m = \{xy \mid \exists uv \in CON \,.\, u \succeq x \succ y \succeq v \wedge \\ (ux \in (\succ - CON) \vee u = x) \wedge (yv \in (\succ - CON) \vee y = v)\}$$

Proof. By Corollary 1, an edge xy is in a minimal full contractor P^- of $\succ$ by CON, if there is CON-detour of at most three edges in $\succ$ in which xy is the only P^--edge. Hence any minimal full contractor is a subset of P^m. Now take every edge xy of P^m and show there is a minimal full contractor of $\succ$ by CON which contains xy. Let $u \succeq x \succ y \succeq v$ for $uv \in CON$. Let us construct a set P' as follows:

$$P' = \begin{cases} \{ux, yv\} & \text{if } u \succ x \wedge y \succ v \\ \{ux\} & \text{if } u \succ x \wedge y = v \\ \{yv\} & \text{if } u = x \wedge y \succ v \\ \emptyset & \text{if } u = x \wedge y = v \end{cases}$$

P' is transitive, $P' \cap CON = \emptyset$, and $P' \subseteq \succ$. Theorem 4 implies that there is a P'-protecting minimal full contractor P^- of $\succ$ by CON. Since P^- protects P', there is a CON-detour in $\succ$ from u to v in which xy is the only P^--edge. This implies that $xy \in P^-$. □

Now consider the case of P^+-protecting full meet contractors. A naive solution is to construct it as the difference of P^m defined above and P^+. However, in the next example we show that such solution does not work in general.

Example 13. *Let a preference relation $\succ$ be a total order of $\{u, x, y, v\}$ (Figure 10). Let also $CON = \{uy, xv\}$ and $P^+ = \{ux\}$. Note that $yv \notin P^+$, and by Theorem 6, $yv \in P^m$. Hence, $yv \in (P^m - P^+)$. However, note that $ux \in P^+$ implies that xy must be a member of every P^+-protecting full contractor in order to disconnect the path from u to y. Hence, there is no CON-detour in which yv is the only edge of the full contractor, and yv is not a member of any P^+-protecting full contractor.*

The next theorem shows how a P^+-protecting full contractor may be constructed. The idea is similar to Theorem 6. However, to construct a full meet contractor, we used the set CON as a common part of all minimal full contractors. In the case of P^+-protecting full meet contractor, a superset C_{P^+} of CON is contained in all of them. Such a set C_{P^+} may be viewed as a union of CON and the set of all edges of $\succ$ that *must be discarded due to the protection of P^+*.

Theorem 7. *Let CON be a finitely stratifiable base contractor of a preference relation $\succ$, and P^+ a transitive relation such that $P^+ \subseteq \succ$ and $P^+ \cap CON = \emptyset$. Then the P^+-protecting full meet contractor of $\succ$ by CON is*

$$P^m_{P^+} = \{xy \mid xy \notin P^+ \wedge \exists uv \in CON \,.\, u \succeq x \succ y \succeq v \wedge \\ (ux \in (\succ - C_{P^+}) \vee u = x) \wedge (yv \in (\succ - C_{P^+}) \vee y = v)\},$$

for

$$C_{P^+} = \{xy \mid \exists uv \in CON \,.\, u \succeq x \succ y \succeq v \wedge (ux \in P^+ \vee u = x) \wedge (yv \in P^+ \vee y = v)\}$$

Proof. First, it is easy to observe that C_{P+} is a subset of any P^+-protecting full contractor of $\succ$ by CON. It is constructed from the edges xy which participate in CON-detours of length at most three where all the other edges have to be protected. Since every CON-detour has to have at least one edge in a full contractor, xy has to be a member of every full contractor.

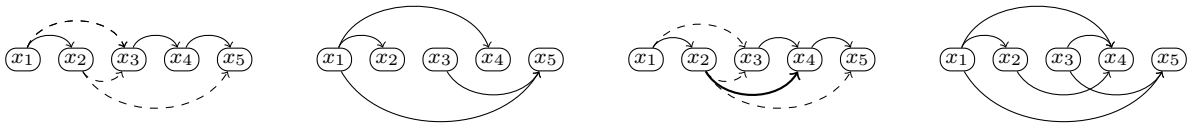
We show that every P^+-protecting minimal full contractor P^- of $\succ$ by CON is a subset of $P^m_{P^+}$. If some $xy \in P^-$, then by Corollary 1 there is an edge $uv \in CON$ such that $u \succeq x \succ y \succeq v$ and $ux, yv \notin P^-$. We show that $xy \in P^m_{P^+}$. That holds if $xy \notin P^+$ (which holds for P^- by definition) and

$$(ux \in (\succ - C_{P^+}) \vee u = x) \wedge (yv \in (\succ - C_{P^+}) \vee y = v)$$

If both $u = x$ and $y = v$ hold then the expression above holds. Now assume $u \succ x$ (the case $y \succ v$ is similar). If $ux \in C_{P^+}$ then, as we showed above, $ux \in P^-$ which is a contradiction. Hence, $ux \in (\succ - C_{P^+})$ and $xy \in P^m_{P^+}$. Finally, $P^- \subseteq P^m_{P^+}$.

Now we show that every $xy \in P^m_{P^+}$ is contained in every P^+-protecting minimal full contractor of $\succ$ by CON. The proof is similar to the proof of Theorem 6. By definition of $P^m_{P^+}$, take xy such that $u \succeq x \succ y \succeq v$. Construct the set P' for xy as in the proof of Theorem 6. We show that for the set $P'' = TC(P^+ \cup P')$ we have $P'' \cap CON = \emptyset$. For the sake of contradiction, assume $P'' \cap CON \neq \emptyset$. This implies that there is a CON-detour consisting of P^+ and P' edges. Having only P^+-edges in the detour contradicts the initial assumption that $P^+ \cap CON = \emptyset$. Having a single edge of P' in the detour implies that the edge (either ux or yv) is in C_{P^+}, which contradicts the definition of $P^m_{P^+}$. Having both ux and yv in the detour implies that $xy \in P^+$ which also contradicts the definition of $P^m_{P^+}$. Hence, $P'' \cap CON = \emptyset$, and by Theorem 5, there is a P''-protecting minimal full contractor P^- of $\succ$ by CON which is also a P^+-protecting minimal full contractor. Since there is a CON-detour in which xy is unprotected by P^-, $xy \in P^-$. □

We note that given the expressions for the meet and P^+-protecting full meet contractors in Theorems 6 and 7, one can easily obtain such contractors for finite and finitely representable relations: by evaluation of a relational algebra query in the former case and by quantifier elimination in the latter case.



(a) $\succ$ and CON (b) $\succ - P^m$ (c) $\succ$, CON, and P^+ (d) $\succ - P^m_{P^+}$

Figure 11: Computing full meet contractor and P^+-protecting full meet contractor

Example 14. *Let a preference relation $\succ$ be a total order of $\{x_1, \ldots, x_5\}$ (Figure 11(a), the transitive edges are omitted for clarity). Let a base contractor CON be $\{x_1x_3, x_2x_3, x_2x_5\}$, and $P^+ = \{x_2x_4\}$.*

A full meet contractor P^m of $\succ$ by CON is $\{x_1x_3, x_2x_3, x_2x_5, x_2x_4, x_3x_4, x_4x_5\}$. The resulting contracted preference relation is shown on Figure 11(b). A P^+-protecting full meet contractor of $\succ$ by CON is $\{x_1x_3, x_2x_3, x_2x_5, x_4x_5\}$. The resulting contracted preference relation is shown on Figure 11(d). Note that C_{P^+} here is $CON \cup \{x_4x_5\}$.

8. Querying with contracted preferences

When dealing with preferences, the two most common tasks are 1) given two tuples, find the more preferred one, and 2) find the most preferred tuples in a set. In this section, we assume that preference and base contractors are represented as *preference formulas*. Out of the two problems above, the first can be solved easily by the evaluation of the corresponding preference formula for the given pair of tuples. To solve the latter problem, the operators of *winnow* [Cho03] and *BMO* [Kie02] are proposed. The winnow picks from a given set of tuples the undominated tuples according to a given preference relation. A special case of the winnow operator is called *skyline* [BKS01]. It operates with preference relations representing *Pareto improvement*. A number of evaluation optimization methods for queries involving winnow have been proposed [Cho07b, HK05, CGGL03, GSG07, PTFS05].

Definition 11. *Let $\mathcal{U}$ be a universe of tuples each having a set of attributes $\mathcal{A}$. Let $\succ$ be a preference relation over $\mathcal{U}$. Then the* winnow operator *is written as $w_\succ(\mathcal{U})$, and for every finite subset r of $\mathcal{U}$:*

$$w_\succ(r) = \{t \in r \mid \neg\exists t' \in r.t' \succ t\}$$

In this section, we show some new techniques which can be used to optimize the evaluation of the winnow operator under contracted preferences.

In user-guided preference modification frameworks [Cho07a, BGS06], it is assumed that users alter their preferences after examining sets of the most preferred tuples returned by winnow. Thus, if preference contraction is incorporated into such frameworks, there is a need to compute winnow under contracted preference relations. Here we show how the evaluation of winnow can be optimized in such cases.

Let $\succ$ be a preference relation, CON be a base contractor of $\succ$, P^- be a full contractor of $\succ$ by CON, and the contracted preference relation $\succ' = (\succ - P^-)$. Denote the set of

the starting and the ending tuples of R-edges for a binary relation R as $S(R)$ and $E(R)$ correspondingly.

$$S(R) = \{x \mid \exists y \; . \; xy \in R\}$$
$$E(R) = \{y \mid \exists x \; . \; xy \in R\}$$

Let us also define the set $M(CON)$ of the tuples which participate in CON-detours in $\succ$

$$M(CON) = \{y \mid \exists x, z \; . \; x \succ y \wedge xz \in CON \wedge y \succeq z\}$$

Assume we also know quantifier-free formulas $F_{S(P^-)}$, $F_{E(P^-)}$, $F_{M(CON)}$, and $F_{S(CON)}$ representing these sets for P^- and CON. Then the following holds.

Proposition 4. *Given a finite set of tuples r*

1. $w_{\succ}(r) \subseteq w_{\succ'}(r)$
2. *If* $\sigma_{F_{S(P^-)}}(w_{\succ}(r)) = \emptyset$, *then* $w_{\succ}(r) = w_{\succ'}(r)$.
3. *If P^- is a prefix full contractor, then* $\sigma_{F_{S(P^-)}}(r) = \sigma_{F_{S(CON)}}(r)$
4. $w_{\succ'}(r) = w_{\succ'}(w_{\succ}(r) \cup \sigma_{F_{E(P^-)}}(r))$

Proof.

1. By definition, $w_{\succ}(r)$ contains the set of undominated tuples w.r.t the preference relation $\succ$. Thus, $\succ' \subset \succ$ implies that if a tuple o was undominated w.r.t $\succ$, it will be undominated w.r.t $\succ'$, too. Hence, $w_{\succ}(r) \subseteq w_{\succ'}(r)$.
2. the SPO of $\succ$ implies that for every tuple o not in $w_{\succ}(r)$, there is a tuple $o' \in w_{\succ}(r)$ such that $o' \succ o$. Hence, if no edges going from $w_{\succ}(r)$ are contracted by P^-, every $o \notin w_{\succ}(r)$ will still be dominated according to $\succ'$.
3. Follows from the definition of the *prefix* contraction.
4. From 1 we know that $w_{\succ}(r) \subseteq w_{\succ'}(r)$. For every tuple $o \in w_{\succ'}(r) - w_{\succ}(r)$, at least one edge going to it in $\succ$ has been contracted by P^-. Thus, $w_{\succ'}(r) \subseteq w_{\succ}(r) \cup \sigma_{F_{E(P^-)}}(r)$ and $w_{\succ'}(r) = w_{\succ'}(w_{\succ}(r) \cup \sigma_{F_{E(P^-)}}(r))$. □

According to Proposition 4, the result of winnow under a contracted preference is always a superset of the result of winnow under the original preference. The second property shows when the contraction does not change the result of winnow. Running the winnow query is generally expensive, thus one can first evaluate $\sigma_{F_{S(P^-)}}$ (or $\sigma_{F_{S(CON)}}$, if P^- is a prefix contraction) over the computed result of the original winnow. If the result is empty, then computing the winnow under the contracted preference relation is not needed.

The last statement of the proposition is useful when the set r is large and thus running $w_{\succ'}$ over the entire set r is expensive. Instead, one can compute $\sigma_{F_{E(P^-)}}(r)$ and then evaluate $w_{\succ'}$ over $(w_{\succ}(r) \cup \sigma_{F_{E(P^-)}}(r))$ (assuming that $w_{\succ}(r)$ is already known).

Example 15. *Let a preference relation $\succ$ be defined by $F_{\succ}(o, o') \equiv o.p < o'.p$, and a base contractor CON of $\succ$ be defined by $F_{CON}(o, o') \equiv o.p = 0 \wedge o'.p = 3$, where p is an Q - attribute. Take set of tuples $r = \{1, 2, 3, 4\}$ in which every tuple has a single attribute p. Then $w_{\succ}(r) = \{1\}$. Take two minimal full contractors P_1^- and P_2^- of $\succ$ by CON defined by the following formulas*

$$F_{P_1^-}(o, o') \equiv o.p = 0 \wedge 0 < o'.p \leq 3$$
$$F_{P_2^-}(o, o') \equiv 0 \leq o.p < 3 \wedge o'.p = 3$$

The corresponding contracted preference relations $\succ_1$ and $\succ_2$ are defined by $F_{\succ_1}(o, o') \equiv F_{\succ}(o, o') \wedge \neg F_{P_1^-}(o, o')$ and $F_{\succ_2}(o, o') \equiv F_{\succ}(o, o') \wedge \neg F_{P_2^-}(o, o')$. The full contractor P_1^- is prefix, thus $F_{S(P_1^-)}(o) \equiv F_{S(CON)} \equiv o.p = 0$. The full contractor P_2^- is not prefix, and $F_{S(P_2^-)}(o) \equiv 0 \leq o.p < 3$.

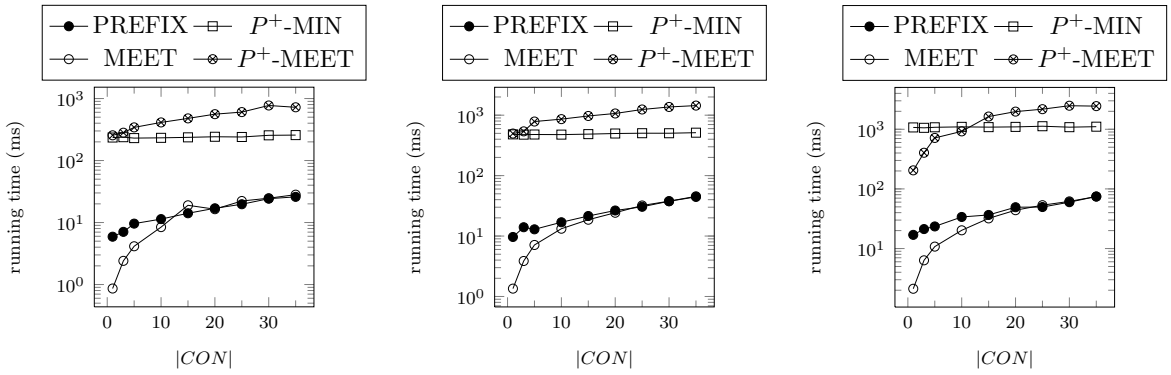
First, $\sigma_{F_{S(P_1^-)}}(w_{\succ}(r)) = \emptyset$ implies $w_{\succ_1}(r) = w_{\succ}(r)$. Second, $\sigma_{F_{S(P_2^-)}}(w_{\succ}(r))$ is not empty and equal to $\{1\}$. Note that $F_{E(P_2^-)}(o) \equiv o.p = 3$. Hence, $\sigma_{F_{E(P_2^-)}}(r) = \{3\}$ and $w_{\succ_2}(r) = w_{\succ_2}(w_{\succ}(r) \cup \sigma_{F_{E(P_2^-)}}(r)) = \{1, 3\}$.

9. Experimental evaluation

In this section, we present the results of experimental evaluation of the preference contraction framework proposed here. We implemented the following operators of preference contraction: prefix contraction (denoted as PREFIX), preference-protecting minimal contraction (P^+-MIN), meet contraction (MEET), and preference-protecting meet contraction (P^+-MEET). PREFIX was implemented using Algorithm 3, P^+-MIN according to Theorem 5, MEET according to Theorem 6, and P^+-MEET according to Theorem 7. We used these operators to contract finite preference relations stored in a database table $R(X, Y)$. The preference relations used in the experiments were finite *skyline preference relations* [BKS01]. Such relations are often used in database applications. We note that such relations are generally not materialized (as database tables) when querying databases with skylines. However, they may be materialized in scenarios of preference elicitation [BGL07]. To generate such relations, we used the NHL 2008 Player Stats dataset [nhl08] of 852 tuples. Each tuple has 18 different attributes out of which we used 5. All algorithms used in the experiments were implemented in Java 6. We ran the experiments on Intel Core 2 Duo CPU 2.1 GHz with 2.0 GB RAM. All tables were stored in a PostgreSQL 8.3 database.

In the first experiment, we modeled the scenario in which a user manually selects preferences to contract. Here we used preference relations consisting of 2000, 3000, and 5000 edges. The sizes of base contractors used here range from 1 to 35 edges. We do not pick more than 35 edges assuming that in this scenario a user unlikely provides a large set of preferences to discard. For every base contractor size, we randomly generated 10 different base contractors and computed the average time spent to compute full contractors and the average size of them. The relations P^+ storing preferences to protect contained 25% of edges of the corresponding preference relation.

Figure 12 shows how the running times of contraction operators depend on the size a preference relation to contract and the size of a base contractor. As we can observe, PREFIX has the best performance among all operators, regardless of the size of the preference relation and the base contractor relation. Note also that the running times of preference-protecting operators are significantly larger then the running times of their unconstrained counterparts. These running times predominantly depend on the time spent to compute the transitive closure of P^+.



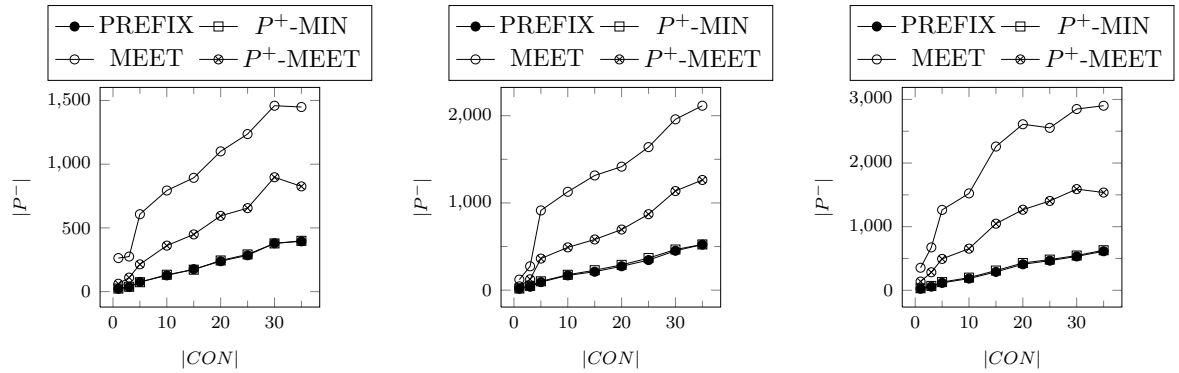
(a) $|\succ| = 2000$ (b) $|\succ| = 3000$ (c) $|\succ| = 5000$

Figure 12: Contraction performance. Small base contractors

Figure 13 shows the dependence of the full contractor size on the size of preference relation and the size of base contractor. For every value of the base contractor size, the charts show the average size of the corresponding full contractor. As we can see, the sizes of minimal full contractors (PREFIX and P^+-MIN) are the least among all full contractors. This supports the intuition that a minimal set of reasons for preferences not to hold is smaller than the set of all such reasons. Another important observation here is that due to the comparatively large size of P^+, the size of a full P^+-protecting meet contractor is generally half the size of the corresponding full meet contractor.

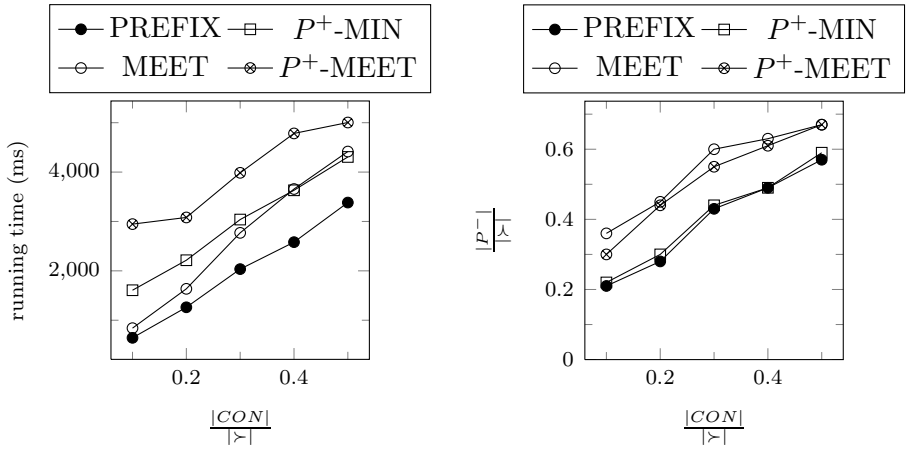
In the next experiment, we assume that base contractors are elicited automatically based on indirect user feedback. Hence, they may be of large size. We construct such relations here from *similar edges*. Two edges xy and $x'y'$ are considered similar if the tuples x, x' and y, $'y$ are similar. We use the cosine similarity measure to compute similarity of tuples. Here we fixed the size of the preference relation to 5000. The sizes of base contractors range from 10% to 50% of preference relation size. The size of every P^+ is 25% of the corresponding preference relation size. Similarly to the previous experiment, we computed the performance of the contraction operators and the sizes of generated full contractors. The results are shown in Figure 14.

First, we note that here the difference between running times of the contraction algorithms is not as large as in the previous experiment. Next, consider the value of the function $aux(CON, P^-) = \frac{|P^-|}{|CON|} - 1$ in this and the previous experiment. $aux(CON, P^-)$ is equal to the average number of edges contracted to contract one edge of CON. Due to the similarity



(a) $|\succ| = 2000$ (b) $|\succ| = 3000$ (c) $|\succ| = 5000$

Figure 13: Full contractor size. Small base contractors



(a) Contraction performance (b) Full contractor size

Figure 14: Large base contractors

of edges in P^-, the value of $aux(CON, P^-)$ is significantly smaller in this experiment than in the previous one. For instance, $aux(CON, P^-)$ according to Figure 14(b) ranges from to 1.1 to 0.1 for PREFIX. For the same algorithm in Figure 13(c), $aux(CON, P^-)$ ranges from 16 to 23.

Note that in all experiments, the time spent to compute any full contractor did not go beyond 5 seconds. If the base contractor is small and preference protection is not used, then these times are even less than 100ms. Hence we conclude that the algorithms we proposed to contract finite relations are efficient and may be used in real-life database applications.

10. Related work

10.1. Relationships with other operators of preference relation change

A number of operators of preference relation change have been proposed so far. An operator of preference revision is defined in [Cho07a]. A preference relation there is *revised* by another preference relation called a *revising relation*. The result of revision is still another preference relation. [Cho07a] defines three semantics of preference revision – union, prioritized, and Pareto – which are different in the way an original and a revising preference relations are composed. For all these semantics, [Cho07a] identifies cases (called 0-*,* 1-*, and* 2-*conflicts*) when the revision fails, i.e., when there is no SPO preference relation satisfying the operator semantics. This work consideres revising preference relations only by preference relations. Although it does not address the problem of discarding subsets of preference relations explicitly, revising a preference relation using Pareto and prioritized revision operators may result in discarding a subset of the original preference relation. It has been shown here that the revised relation is an SPO for limited classes of the composed relations.

Another operator of preference relation change is defined in [BGS06]. This work deals with a special class of preference relations called *skyline* [BKS01]. Preference relations in [BGS06] are changed by *equivalence relations*. In particular, a modified preference relation is an extension of the original relation in which specified tuples are *equivalent*. This change operator is defined for only those tuples which are *incomparable* or already *equivalent* according to the original preference relation. This preference change operator only adds new edges to the original preference relation, and thus, preference relation contraction cannot be expressed using this operator.

In [MC08], we introduced the operation of minimal preference contraction for preference relations. We studied properties of this operation and proposed algorithms for computing full contractors and preference-protecting full contractors for finitely stratifiable base contractors. In the current paper, we generalize this approach and we develop a method of checking the finite stratifiability property for finitely representable base contractors. We introduce the operations of meet and meet preference-protecting contraction, and propose methods for computing them. We also provide experimental evaluation of the framework and a comprehensive discussion of related work.

10.2. Relationships with the belief revision theory

Preferences can be considered as a special form of human *beliefs*, and thus their change may be modeled in the context of the belief change theory. The approach here is to represent beliefs as truth-functional logical sentences. A *belief set* is a set of the sentences that are believed by an agent. A common assumption is that belief sets are closed under logical consequence. The most common operators of belief set change are *revision* and *contraction* [AGM85]. A number of versions of those operators have been proposed [Han98] to capture various real life scenarios.

This approach is quite different from the preference relation approach. First, the language of truth functional sentences is rich and allows for rather complex statements about preferences: conditional preferences ($a > b \rightarrow c > d$), ambiguous preferences ($a > b \vee c > d$)

etc. In contrast to that, preferences in the preference relation framework used in this paper are certain: given a preference relation $\succ$, it is only possible to check if a tuple is preferred or not to another tuple. Another important difference of these two frameworks is that the belief revision theory exploits the open-world assumption, while the preference relation framework uses the closed-world assumption. In addition to that, belief revision is generally applicable in the context of finite domains. However, the algorithms we have proposed here can be applied to finite and infinite preference relations.

10.3. Relationships with the preference state framework

Another preference representation and change framework close to the belief revision theory is the *preference state* framework [Han95]. As in belief revision, a preference state is a logically closed sets of sentences describing preferences of an agent. However, every preference state has an underlying set of preference *relations*. The connection between states and relations is as follows. A preference relation (which is an order of tuples) is an unambiguous description of an agent preference. A preference relation induces a set of logical sentences which describe the relations. However, it is not always the case that people's preferences are unambiguous. Hence, every preference *state* is associated with a *set* of possible preference relations.

Here we show an adaptation of the preference state framework to the preference relation framework. As a result, we obtain a framework that encompasses preference contraction and restricted preference revision.

Definition 12. *An* alternative *is an element of* $\mathcal{U}$*. Nonempty subsets of* $\mathcal{U}$ *are called* sets of alternatives. *The tuple language* $\mathcal{L}_{\mathcal{U}}$ *is defined as*

- *if* $X, Y \in \mathcal{U}$ *then* $X > Y \in L_{\mathcal{U}}$
- *if* $X > Y \in L_{\mathcal{U}}$ *then* $\neg(X > Y) \in L_{\mathcal{U}}$.

A subset of $\mathcal{L}_{\mathcal{U}}$ is called a *restricted preference set*. The language defined above is a very restricted version of the language in [Han95] since the only Boolean operator allowed is negation. Throughout the discussion, we assume that the set of alternatives is fixed to a subset $\mathcal{U}_r$ of $\mathcal{U}$.

Definition 13. *Let* R *be a subset of* $\mathcal{U}_r \times \mathcal{U}_r$*. The set* $[R]$ *of sentences is defined as follows:*

- $x > y \in [R]$ *iff* $xy \in R$
- $\neg(x > y) \in [R]$ *iff* $x, y \in \mathcal{U}_r$ *and* $x > y \notin [R]$

Definition 14. *A binary relation* $R \subset \mathcal{U}_r \times \mathcal{U}_r$ *is a* restricted preference model *iff it is a strict partial order. Given a restricted preference model* R*, the corresponding* $[R]$ *is called a* restricted preference state.

In contrast to the definition above, the preference model in [Han95] is defined as a *set* of SPO relations, and a preference state is an intersection of $[R]$ for all members R of the corresponding preference model.

We define two operators of change of restricted preference states: revision and contraction. Restricted states here are changed by sets of statements. In [Han95], a change of a preference state by a set of sentences is defined as the corresponding change by the conjunction of the corresponding statements. Moreover, change by any set of sentences is allowed. In the adaptation of that framework we define here, conjunctions of statements are not a part of the language. Moreover, preference revision [Cho07a] only allows for adding new preferences, and preference relation contraction we have proposed in this paper allows only discarding existing preferences. Here we aim to define the operator of restricted preference set revision which captures the semantics of those two operators.

Definition 15. *A restricted preference set S is called* positive *iff all sentences it contains are in form $A > B$ for some $A, B \in \mathcal{U}_r$. Analogously, S is* negative *iff it only contains sentences in form $\neg(A > B)$ for some $A, B \in \mathcal{U}_r$.*

A restricted preference set is a complement *of S (denoted as $\overline{S}$) if for all $A, B \in \mathcal{U}_r$, $A > B \in S$ iff $\neg(A > B) \in S$ and $\neg(A > B) \in S$ iff $A > B \in S$.*

A relation R_S is a minimal representation *of a restricted preference state S iff R_S is a minimal relation such that $S \subseteq [R_S]$.*

Positive and negative restricted preference sets are used to change restricted preference states. Intuitively, a positive preference set represents the existence of preferences while a negative set represents a lack of preferences.

Definition 16. *Let R be a restricted preference model. Then the operator $*$ on R is a* restricted preference revision on R *if and only if for all positive/negative restricted preference sets S, $R * S = \cap\{R'\}$ for all R' such that*

1. *$S \subseteq [R']$*
2. *R' is an SPO*
3. *there is no SPO R'' with $S \subseteq [R'']$ such that $R \subseteq R'' \subset R'$ (if S is positive) or $R' \subset R'' \subseteq R$ (if S is negative).*

The last condition in the definition above expresses the minimality of restricted preference state change. This condition is different for positive and negative sets: when we add positive statements, we do not want to discard any existing positive sentences, and when negative statements are added, no new positive sentences should be added. The restricted preference revision operator defined above is different from preference state revision in [Han95]. First, preference state revision allows for revision by (finite) sets of arbitrary sentences, not only positive and negative sentences, as here. Second, the minimality condition here is defined using set containment while in [Han95] it is defined as a function of symmetric set difference of the original preference relations and R'. As a result, revising by preference state by a positive/negative sentence may result in losing an existing positive/negative sentence. The

last difference is based on preference state representation: the result of preference revision in [Han95] is a union of relations R'' while in our case it is an intersection.

Below we define the operator of contraction for restricted preference states which is similar to the contraction of preference states.

Definition 17. *Let R be a restricted preference model. Then the operator $\div$ on R is* restricted preference contraction on R *if and only if for all positive/negative restricted preference sets S, $R \div S = R * \overline{S}$.*

Given the operators on restricted preference states we have defined here, their relationships with the preference change framework are straightforward.

Proposition 5. *Let R be a restricted preference model, S be a positive or negative restricted preference set, and R_S be a minimal representation of S. Then $R * S$ is*

1. *$\emptyset$, if S is a positive restricted preference set and $R \cup R_S$ has cyclic path,*
2. *$TC(R \cup R_S)$, if S is a positive restricted preference set and $R \cup R_S$ has no cyclic paths,*
3. *$\cap\{R - P^- \mid P^-$ is a minimal full contractor of R by $\overline{R_S}\}$, if S is a negative restricted preference set,*

where TC is the transitive closure operator.

Proof. When a restricted preference model is revised by a positive preference set, the resulting relation $R * S$ is the intersection of all minimal SPO extensions R' of R and R_S (i.e., R' has to contain an edge from A to B if $A > B \in S$). Such an extension R' does not exist if there is an cyclic path in $R \cup R_S$. However, if no cyclic paths exist, then there is only one such a minimal extension R' which is equal to the transitive closure of $R \cup R_S$. Hence, $R * S = TC(R \cup R_S)$. We note that this result is equivalent to the result of the *union preference revision* [Cho07a].

When a restricted preference model is revised by a negative preference set, the resulting relation $R * S$ has to be a subset of R. Moreover, for all $\neg(A > B) \in S$, there should be no edge from A to B in $R * S$. Hence, $R * S$ is an intersection of minimally contracted R by $R_{\overline{S}}$, which is a result of the full meet contraction of R by $R_{\overline{S}}$. □

Below we list some properties of the revision and contraction operators of restricted preference states.

Proposition 6. *Let R be a restricted preference model and S be a positive/negative restricted preference set. Then*

1. *$R * S$ is an SPO (closure)*
2. *$S \subseteq [R * S]$ unless S is positive and $R_S \cup R$ has a cyclic path (limited success)*
3. *If $S \subseteq [R]$, then $R = R * S$ (vacuity)*

PROOF. All the properties here follow from Proposition 5. Namely, property 1 follows from the fact that the result of $R * S$ is an SPO in every case of Proposition 5. Property 2 follows from Proposition 5 and the definition of $[R * S]$. Property 3 follows from Proposition 5 and 1) $S \subseteq [R]$ implies $R_S \subseteq R$ (if S is positive), and 2) a minimally contracted preference relation is equal to itself if contracted by non-existent edges (if S is negative). □

Proposition 7. *Let R be a restricted preference model and S be a restricted positive/negative preference set. Then*

1. *$R \div S$ is an SPO (closure)*
2. *$S \subseteq [R \div S]$ unless S is negative and $R_{\overline{S}} \cup R$ has a cyclic path (limited success)*
3. *If $S \cap [R] = \emptyset$, then $R = R \div S$ (vacuity)*
4. *$R * S = (R \div \overline{S}) * S$ unless S is positive and $R_S \cup R$ has a cyclic path (limited Levi identity)*
5. *$R \div S = R * \overline{S}$ (Harper identity, by definition)*

PROOF. Properties 1, 2, and 3 follow from Proposition 6. Property 4 follows from the fact that $R \div \overline{S} = R * S$ by definition, and Proposition 6 implies $R * S = (R * S) * S$ when either S is negative or S is positive but $R_S \cup R$ has no cyclic path. □

An important difference between the restricted preference-set change operators and the corresponding change operators from [Han95] is that the restricted versions are not always successful (property 2 in Proposition 5), and Levi identity holds for a certain class of restricted preference sets. In addition to that, the operator of preference set contraction in [Han95] has the property of inclusion ($R \subseteq R \div S$) and recovery (if $S \subseteq [R]$, then $R = (R \div S) * S$). As for the restricted framework defined here, inclusion does not hold due to the representation of a preference model as a single SPO relation. Recovery does not hold here due to the restrictions to the language (namely, not allowing disjunction of sentences).

We note that one of the main targets of our current work was development of an efficient and practical approach of contracting preference relations in the binary relation framework, in the finite and the finitely representable cases. In addition to the defining semantics of preference contraction operators, we have also developed a set of algorithms which can be used to compute contractions. We have tested them on real-life data and demostrated their efficiency. In contrast, [Han95] focuses more on semantical aspects of preference change and does not address computational issues of preference change operators. In particular, finite representability is not addressed.

10.4. Other related frameworks

An approach of preference change is proposed in [CP06]. Preferences here are changed via *interactive example critiques*. This paper identified three types of common critique models: similarity based, quality based, and quantity based. However, no formal framework is provided here. [Fre04] describes revision of *rational* preference relations over propositional formulas. The revision operator proposed here satisfies the postulates of success and minimal

change. The author shows that the proposed techniques work in case of revision by a single statement and can be extended to allow revisions by multiple statements.

[DLSL99] proposes algorithms of incremental maintenance of the transitive closure of graphs using relational algebra. The graph modification operations are edge insertion and deletion. Transitive graphs in [DLSL99] consist of two kinds of edges: the edges of the original graph and the edges induced by its transitive closure. When an edge xy of the original graph is contracted, the algorithm also deletes all the transitive edges uv such that all the paths from u to v in the original graph go through xy. As a result, such contraction is not minimal according to our definition of minimality. Moreover, [DLSL99] considers only finite graphs, whereas our algorithms can work with infinite relations.

11. Conclusions and future work

In this paper, we have presented an approach to contracting preference relations. We have considered several operators of preference contraction: minimal preference contraction, minimal preference-preserving contraction, and (preference protecting) meet contraction inspired by different scenarios of cautious preference change. We have proposed algorithms and techniques of computing contracted preference relations for a class of finite and finitely representable relations. We have introduced some techniques of optimizing preference queries in the presence of contraction. We have also evaluated the proposed algorithms experimentally and showed that they can be used in real-life database applications.

We have shown how preference contraction can be evaluated for a special class of finitely stratifiable base contractors. One of the areas of our future work is to relax that property and consider more general base contractors.

An interesting direction of future work is to design an operator of generalized preference relation change that allows to change preference relations by discarding existing as well as adding new preferences at the same time. The current approaches of preference relation change are restricted to only one type of change.

As we showed in the discussion of related work, the existing preference revision approach [Cho07a] fails to work in the presence of conflicts (cycles). A promising direction here is to use the preference contraction operators presented here to resolve such conflicts.

In this paper, we assume that the relations defining the preferences to discard are explicitly formulated by the user. However, such an assumption hardly works in practical scenarios of preference change: formulating such a relation requires a full knowledge of his or her preferences, which may not be the case. Hence, a promising direction is to perform interactive preference contraction or change.

Appendix A

Theorem 3. (Checking finite stratifiability property). *Let F_R be an ERO-formula in DNF, representing an SPO relation R, of the following form*

$$F_R(o, o') = F_{R_1}(o, o') \vee \ldots \vee F_{R_l}(o, o'),$$

where F_{R_i} is a conjunction of atomic formulas. Then checking if there is a constant k such that the length of all R-paths is at most k can be done by a single evaluation of QE over a formula of size linear in $|F_R|$.

Let R_i be a binary relation represented by the formula F_{R_i} for all $i \in [1, l]$. We split the proof of Theorem 3 into several lemmas. In Lemma 4, we show that the length of all R-paths is bounded by a constant if and only if the length of all R_i-paths is bounded by a constant for every disjunct F_{R_i} of F_R. Lemma 5 shows that the length of all R_i-paths is bounded by a constant if and only if there is a bound on the length of all paths induced by a relation represented by at least one conjunct of F_{R_i}. In Lemma 6, we show how to check if the length of all paths induced by a conjunct of F_{R_i} is bounded.

To prove the first lemma, we use the following idea. Let a sequence $S = (o_1, \ldots, o_n)$ of $n \geq 2$ tuples be an R-sequence, i.e.,

$$(o_1, o_2), \ldots, (o_{n-1}, o_n) \in R \tag{1}$$

The transitivity of R implies that there is an R-edge from o_1 to all other tuples in S, i.e.,

$$(o_1, o_2), \ldots, (o_1, o_n) \in R \tag{2}$$

Note that (2) contains only edges started by o_1. Since $R = \cup_{i=1}^{l} R_i$, for every R-edge in (2), there is $i \in [1, l]$ such that it is also an R_i-edge. Let R_j for some $j \in [1, l]$ be such that the number of R_j-edges in (2) is maximum. Such R_j is called *a major component of S*. Let the sequence S' consist of the end nodes of all these R_j-edges in the order they appear in S. Such S' is called *a major subsequence of S*.

Observation 1. *Let S be an R-sequence, R_{i^*} a major component of S, and S' be the corresponding major subsequence of S. Then*

1. *S' is an R-sequence*
2. *if the length of S is n, then the length of S' is at least $\lceil \frac{n-1}{l} \rceil$*

The first fact of Observation 1 follows from transitivity of R, and the second fact follows from the definition of major subsequence. Note that a major subsequence is an R-sequence too. Hence, if it has at least two tuples, we can construct its major subsequence.

Observation 2. *Let $S_0, \ldots, S_t$ be R-sequences such that for all $i \in [1, t]$, S_i is a major subsequence of S_{i-1} with the corresponding major components R_{j_i}. Let o, o' be the first tuples of S_1 and S_t correspondingly. Then $R_{j_1}(o, o')$.*

Observation 2 follows from the definition of major subsequence.

Example 16. *Let $S_0 = (x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{11}, x_{12})$ be an R-sequences. Figure 15 illustrates possible construction of a major subsequence S_1 of S_0, a major subsequence S_2 of S_1, and a major subsequence S_3 of S_2. The edges on Figure 15 correspond to the major-component edges. In every sequence, a node is dark if it is in the major subsequence of the sequence. Note that S_3 does not have a major subsequence because a subsequence has to have at least two nodes.*

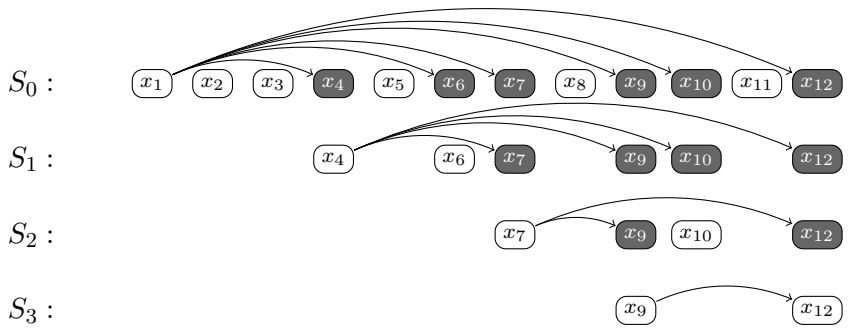


Figure 15: Major subsequences

Lemma 4. *There is a constant bounding the length of all R-paths if and only if for all $i \in [1, l]$, there is a constant bounding the length of all R_i-paths.*

Proof. In the case when $l = 1$, the lemma trivially holds. Further we assume $l > 1$.
$(\Rightarrow)$ If for some $i \in [1, l]$, the length of R_i-paths cannot be bounded, neither can the length of R-paths.
$(\Leftarrow)$ Assume that for all $i \in [1, l]$, all R_i-paths are of length at most k. Show that the length of all R-paths is not more than $\sum_{i=1}^{(k+2)l+1} l^i - 2$. For the sake of contradiction, let there be an R-path of length $\sum_{i=1}^{(k+2)l+1} l^i - 1$. Let S_0 be the corresponding R-sequence. The size of S_0 is $\sum_{i=0}^{(k+2)l+1} l^i$. Let S_1 be a major subsequence of S_0. By Observation 1, S_1 is also an R-sequence, and its length is at least $\sum_{i=0}^{(k+2)l} l^i$. Following that logic, let S_t be a major subsequence of S_{t-1} with the corresponding major component $R_{j_{t-1}}$. The size of S_t is at least $\sum_{i=0}^{(k+2)l-t+1} l^i$. Such computation may continue while the size of S_t is greater than one, i.e., while $t \leq (k+2)l$. Let the major components of $S_1, \ldots, S_{(k+2)l}$ be $R_{j_1}, \ldots, R_{j_{(k+2)l}}$ correspondingly. Note that there are at most l possible different major components. Thus, at least $k+2$ major components in $R_{j_1}, \ldots, R_{j_{(k+2)l}}$ are the same. Let us denote the first $k+2$ of them as $R_{t_1}, \ldots, R_{t_{k+2}}$ and the tuples which start the corresponding major sequences as $o_{t_1}, \ldots, o_{t_{k+2}}$. By Observation 2,

$$R_{t_1}(o_{t_1}, o_{t_2}) \wedge R_{t_2}(o_{t_2}, o_{t_3}) \wedge \ldots \wedge R_{t_{k+1}}(o_{t_{k+1}}, o_{t_{k+2}})$$

Since all $R_{t_1}, \ldots, R_{t_{k+2}}$ are the same, the expression above implies that there is an R_i-path of length $k+1$ for some $i \in [1, l]$ which is a contradiction. □

In Lemma 4, we showed that the problem of checking the bounded-length property of all R-paths can be reduced to the problem of testing the same property for R_i-paths. Note that R_i is represented by a formula F_{R_i} which is a conjunction of atomic formulas. Let the set of all attributes which are present in the formula F_{R_i} be defined as $\mathcal{A}_{F_{R_i}}$. Then F_{R_i} can be represented as

$$F_{R_i}(o, o') = \bigwedge_{A \in \mathcal{A}_{F_{R_i}}} \lambda_A(o, o'),$$

where $\lambda_A(o, o')$ is a conjunction of all atomic formulas in which the attribute A is used. Note that the structure of the preference formula language implies that *every atomic formula belongs to exactly one* λ_A.

Denote the relation represented by λ_A as Λ_A. In the next lemma, we show that the problem of checking the finite stratifiability property of all R_i-paths can be reduced to the same problem for Λ_A-paths.

Lemma 5. *There is a constant bounding the length of all R_i-paths if and only if for some $A \in \mathcal{A}_{F_{R_i}}$, there is a constant bounding the length of all Λ_A-paths.*

Proof.
$\Leftarrow$ Let for every k, there be an R_i-path of length at least k

$$R_i(o_1, o_2) \wedge R_i(o_2, o_3) \wedge \ldots \wedge R_i(o_k, o_{k+1})$$

Then for all $A \in \mathcal{A}_{F_{R_i}}$, we have a Λ_A-path of length at least k

$$\Lambda_A(o_1, o_2) \wedge \Lambda_A(o_2, o_3) \wedge \ldots \wedge \Lambda_A(o_k, o_{k+1})$$

$\Rightarrow$ Let for every k and $A \in \mathcal{A}_{F_{R_i}}$, there be an Λ_A-path of length at least k

$$\Lambda_A(o_1^A, o_2^A) \wedge \Lambda_A(o_2^A, o_3^A) \wedge \ldots \wedge \Lambda_A(o_k^A, o_{k+1}^A)$$

Construct a sequence of tuples $(o_1, o_2, o_3, \ldots)$ as follows. Let $o_j.A = o_j^A.A$ if $A \in \mathcal{A}_{F_{R_i}}$. Otherwise, let $o_j.A$ be any value from the domain $\mathcal{D}_A$ of A. Clearly, the following R_i-path is of length at least k

$$R_i(o_1, o_2) \wedge R_i(o_2, o_3) \wedge \ldots \wedge R_i(o_k, o_{k+1})$$

□

Lemma 6. *There is a constant bounding the length of all Λ_A-paths if and only if there is no Λ_A-path of length three, i.e.,*

$$\neg\exists o_1, o_2, o_3, o_4 \in \mathcal{U} \ . \ \Lambda_A(o_1, o_2) \wedge \Lambda_A(o_2, o_3) \wedge \Lambda_A(o_3, o_4)$$

Proof.
$\Leftarrow$ If for every constant k, there is a Λ_A-path of length at least k, then there is a Λ_A-path of length three.
$\Rightarrow$ If Λ_A is unsatisfiable, then there are no Λ_A-paths. Thus, we assume that Λ_A is satisfiable. Based on the preference formula language, the formula $\lambda_A(o, o')$ can be split into at most three conjunctive formulas:

1. ϕ_L: a conjunction of all atomic formulas $o.A\theta c$,
2. ϕ_R: a conjunction of all atomic formulas $o'.A\theta c$,
3. ϕ_M: a conjunction of all atomic formulas $o.A\theta o'.A$

for $\theta \in \{=, \neq, <, >\}$ and a C - or Q -constant c. Any of these three formulas may be missing because λ_A may not containt atomic formulas of the specified type. ϕ_L and ϕ_R capture the range of the left and the right argument in λ_A, correspondingly, and ϕ_M constrains their relationship.

Here we assume that A is a Q -attribute, and the case of C -attributes is similar. Note that if ϕ_L is defined, then the range r_L of ϕ_L is 1) an open rational number interval with a finite number of holes (due to possible atomic formulas $o.A \neq c$), or 2) a single rational value (due to the formula $o.A = c$). If ϕ_L is undefined, then r_L is the entire set of rational numbers. Thus, the the number of distinct elements $|r_L|$ in r_L is either ∞ or 1. The same holds for the number of distinct elements $|r_R|$ in r_R. Hence for our class of formulas, $|r_L \cap r_R| \in \{1, \infty\}$.

Now consider the structure of ϕ_M. If ϕ_M is undefined, then $|r_L \cap r_R| > 0$ implies that there are Λ_A-paths of length at least k for every k, consisting of tuples whose A-values are arbitrary elements of $r_L \cap r_R$. If "$o.A = o'.A$" $\in \phi_M$, then no other atomic formula is in ϕ_M (otherwise, Λ_A is unsatisfiable). Since $|r_L \cap r_R| > 0$, Λ_A-paths of length at least k for every k can be constructed of tuples with the value of A all equal to any member of $r_L \cap r_R$. If "$o.A > o'.A$" $\in \phi_M$, then "$o.A = o'.A$", "$o.A < o'.A$" $\notin \phi_M$ (otherwise λ_A is unsatisfiable). However, "$o.A \neq o'.A$" may be in ϕ_M and is implied by "$o.A > o'.A$" $\in \phi_M$ so can be dropped. The existence of a Λ_A-path of length three implies that $|r_L \cap r_R| > 1$ and thus $|r_L \cap r_R| = \infty$. Hence there are Λ_A-paths of length at least k for every k. The case of "$o.A < o'.A$" $\in \phi_M$ is analogous. The last case is when only "$o.A \neq o'.A$". The existence of a Λ_A-path of the length three implies that there are two different values $c_1, c_2 \in r_L \cap r_R$. Hence, Λ_A-paths of length at least k for every k can be constructed by taking any sequence of tuples in which the value of A of every even tuple is c_1 and of every odd tuple is c_2. □

Proof of Theorem 3. Here we show how to construct a formula which is true iff there is a constant k such that the length of all R-paths is bounded by k. By Lemma 4, such a formula can be written as a conjunction of l formulas each of which represents the fact that the length of all R_i-paths is bounded. By Lemma 5, such a formula can be written as a disjunction of formulas each of which represents the fact that the length of all Λ_A-paths is bounded. By Lemma 6, such formulas are of size linear in the size of Λ_A. Hence, the resulting formula is linear in the size of F_R. Due to the construction in Lemma 6, the formula has quantifiers. They can be eliminated using QE. □

References

[AGM85] Carlos E. Alchourron, Peter Gardenfors, and David Makinson. On the logic of theory change: Partial meet contraction and revision functions. *The Journal of Symbolic Logic*, 50(2):510–530, 1985.

[BBD+04] C. Boutilier, R. Brafman, C. Domshlak, H. Hoos, and D. Poole. CP-nets: a tool for representing and reasoning with conditional ceteris paribus preference statements. *Journal of Artificial Intelligence Research (JAIR)*, 21:135–191, 2004.

[BGL07] Wolf-Tilo Balke, Ulrich Güntzer, and Christoph Lofi. Eliciting matters - controlling skyline sizes by incremental integration of user preferences. In *DASFAA*, pages 551–562. Springer, April 2007.

[BGS06] Wolf-Tilo Balke, Ulrich Guntzer, and Wolf Siberski. Exploiting indifference for customization of partial order skylines. In *Proc. IDEAS '06*, pages 80–88, 2006.

[BKS01] Stephan Börzsönyi, Donald Kossmann, and Konrad Stocker. The skyline operator. In *Proceedings of the 17th International Conference on Data Engineering*, pages 421–430, Washington, DC, USA, 2001. IEEE Computer Society.

[CGGL03] Jan Chomicki, Parke Godfrey, Jarek Gryz, and Dongming Liang. Skyline with presorting. In *ICDE*, pages 717–816. IEEE Computer Society, March 2003.

[Cho03] Jan Chomicki. Preference formulas in relational queries. *ACM Trans. Database Syst.*, 28(4):427–466, 2003.

[Cho07a] J. Chomicki. Database querying under changing preferences. *Annals of Mathematics and Artificial Intelligence*, 50(1-2):79–109, 2007.

[Cho07b] J. Chomicki. Semantic optimization techniques for preference queries. *Inf.Syst.(IS)*, 32(5):670–684, 2007.

[CLRS01] T. Cormen, C. Leiserson, R. Rivest, and C. Stein. *Introduction to Algorithms, Second Edition*. The MIT Press, September 2001.

[CP06] Li Chen and Pearl Pu. Evaluating critiquing-based recommender agents. In *Proceedings of AAAI-2006*, Boston, Massachusetts, USA, July 2006. AAAI Press.

[DLSL99] G. Dong, L. Libkin, J. Su, and L.Wong. Maintaining the transitive closure of graphs in SQL. *Int. Journal of Information Technology*, 5:46–78, 1999.

[Doy04] Jon Doyle. Prospects for preferences. *Computational Intelligence*, 20(2):111–136, 2004.

[Fis70] P. C. Fishburn. *Utility theory for decision-making*. Wiley, New York, 1970.

[Fre04] Michael Freund. On the revision of preferences and rational inference processes. *Artificial Intelligence*, 152(1):105–137, 2004.

[GSG07] Parke Godfrey, Ryan Shipley, and Jarek Gryz. Algorithms and analyses for maximal vector computation. *VLDB Journal*, 16(1):5–28, 2007.

[Han95] S. O. Hansson. Changes in preference. *Theory and Decision*, 38(1):1–28, 1995.

[Han98] Sven Ove Hansson. *Handbook of Defeasible Reasoning and Uncertainty Management Systems*, volume 3: Belief Change, chapter Revision of belief sets and belief bases, pages 17–75. Kluwer Academic Publishers, Dordrecht, October 1998.

[HGY06] Sven Ove Hansson and Till Grüne-Yanoff. Preferences. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*, page http://plato.stanford.edu/archives/win2006/entries/preferences/. 2006.

[HK05] B. Hafenrichter and W. Kießling. Optimization of relational preference queries. In *ADC '05: Proceedings of the 16th Australasian Database Conference*, pages 175–184, 2005.

[Kie02] Werner Kießling. Foundations of preferences in database systems. In *Proceedings of 28th International Conference on Very Large Data Bases*, pages 311–322, Hong Kong, China, August 2002. Morgan Kaufmann.

[KKR95] P. C. Kanellakis, G. M. Kuper, and P. Z. Revesz. Constraint query languages. *Journal of Computer and System Sciences*, pages 26–52, 1995.

[MC08] D. Mindolin and J. Chomicki. Minimal contraction of preference relations. In *Proceedings of AAAI-2008*, pages 492–497, Chicago, Illinois, USA, July 2008. AAAI Press.

[nhl08] NHL.com Stats, 2008. `http://www.nhl.com/ice/playerstats.htm`.

[PTFS05] Dimitris Papadias, Yufei Tao, Greg Fu, and Bernhard Seeger. Progressive skyline computation in database systems. *ACM Transactions on Database Systems (TODS)*, 30(1):41–82, 2005.

[RG02] R. Ramakrishnan and J. Gehrke. *Database Management Systems*. McGraw-Hill Science/Engineering/Math, August 2002.